1984

Multilevel security for the Integrated Software System mail application.

Wyatt, Robert Walter

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/19458

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

MULTILEVEL SECURITY FOR THE INTEGRATED SOFTWARE SYSTEM MAIL APPLICATION

by

Robert W. Wyatt

March 1984

Thesis Advisor: Dushan Z. Badal

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Multilevel Security for the Integrated Software System Mail Application		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March, 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Robert W. Wyatt		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE March, 1984
		13. NUMBER OF PAGES 136
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Security, Multilevel Security, Electronic Mail, Relational Database Model

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In order to avoid the development of the entire conceptual design of a multilevel secure electronic mail application, an approach is taken to develop the design theough the integration of multilevel security featues into an existing conceptual design. The conceptual design of the electronic mail application of the Integrated Software System (ISS) is used as the source of application specific functions. Thus the aim of the thesis (Continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT (Continued)

is the conceptual design of those features which would make the ISS electronic mail application multilevel secure.
The first section of the thesis explores those issues and areas of work which impact on the design of the security features.
The second section develops the conceptual design of the security features. During the design, the author establishes the attributes necessary to support multilevel secure access mediation, defines modularization which supports and enhances security, and defines the user interface required by the modularization.

Approved for public release, distribution unlimited

Multilevel Security for the Integrated Software System Mail Application

by

Robert Walter Wyatt
Civilian, United States Department of Defense
B.A., College of William and Mary, 1974
B.S., University College/University of Maryland, 1981

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 1984

ABSTRACT

In order to avoid the development of the entire conceptual design of a multilevel secure electronic mail application, an approach is taken to develop the design through the integration of multilevel security features into an existing conceptual design. The conceptual design of the electronic mail application of the Integrated Software System (ISS) is used as the source of application specific functions. Thus the aim of the thesis is the conceptual design of those features which would make the ISS electronic mail application multilevel secure.

The first section of the thesis explores those issues and areas of work which impact on the design of the security features. The second section develops the conceptual design of the security features. During the design, the author establishes the attributes necessary to support multilevel secure access mediation, defines a modularization which supports and enhances security, and defines the user interface required by the modularization.

TABLE OF CONTENTS

I. INTRODUCTION

Although computers have long been used to process sensitive military information, the marriage of classified information and computers has not been as blissful as hoped. Part of the reason for this stems from the lack of concern for computer security during the design and implementation of both computer systems and their software, resulting in inadequate internal security controls. In the absence of such internal controls, the Department of Defense (DoD) has had to establish extensive externally based controls aimed at compensating for the lack of internal ones. These compensatory efforts are evidenced in the appearance of such modes of operations as dedicated, system high, and periods processing. Such modes of operation do heighten security but do not completely compensate for proper internal security controls.

An increased effort has been made in recent years to design and implement computer systems and software with sufficient internal security controls to provide proper protection of classified information. The end goal of such an effort is the development of systems permitting sensitive information of varying levels of classification and type to be stored simultaneously on the system and selectively accessed by users having varying security clearances and

access capabilities. Such systems would be considered to be multilevel secure.

It is the application of this multilevel security concept to the design of an electronic mail system that is the topic of this thesis. In order to avoid the necessity of developing both electronic mail functions and security functions, the existing design of the Integrated Software System's (ISS) [Ref. 1] electronic mail application will be used to provide the basic electronic mail functions. Thus, the scope of this thesis is the conceptual design of those features which would make the ISS electronic mail application multilevel secure.

The thesis can be divided roughly into two interrelated sections. The first section provides the reader with a foundation of terminology and concepts underlying the conceptual design by exploring those issues and areas of work which have had an impact on the design decisions made in the second section. Some of the issues covered are the development of the DoD requirement for multilevel secure systems, attempts to define the requirements of multilevel security, and the role that formal models play in the development of multilevel secure systems. Related work topics include the ISS, secure military message systems, modularization, and database security.

The second section of the thesis presents the conceptual design of the multilevel security features for the

electronic mail application. The design has three primary objectives. Since the selected application is based on the relational database model, the first objective is to establish those attributes necessary to fully support the access mediation required of a multilevel secure electronic mail application. The second objective is to define a modularization of information and functions which supports and enhances the security aspects of the application. Finally, an attempt will be made to define the user interface required by the proposed modularization.

II. MULTILEVEL SECURITY

A. INTRODUCTION

This chapter introduces the issues of multilevel security and work in the area of developing multilevel secure systems. Section B discusses the historical development of the multilevel security issue and it elaborates on the Department of Defense's (DoD) requirements for multilevel secure systems. Section C presents a view of efforts to refine the requirements of a multilevel secure system. Section D, describes the role of formal models in the development of secure systems.

B. DOD REQUIREMENTS FOR MULTILEVEL SECURITY

When one considers DoD requirements for multilevel security, two facts must be initially stated. The first is security. It is generally accepted that controls must be established to restrict the handling of sensitive military information. The exact controls implemented depend on the classification (to include the level of classification and any compartments and/or caveats) and the security policy in effect at the time in question. The second fact is the DoD's need for computers. It has been shown that the U.S. falls short of quantitative superiority in a number of defense forces. Computers, however, give the DoD a qualitative factor which helps to overcome the quantitative

shortcomings. In the command, control, and communications (C3) field alone, it has been noted that "good C3 [command, control, and communications] capabilities can double or triple force effectiveness; conversely, ineffective C3 is certain to jeopardize or deny the objective sought." [Ref. 2: pp. 17-18]

Neither of the two facts taken alone have posed much problem in the past. Well established procedures exist which can effectively control the dissemination of classified material via the traditional medium of paper. [Ref. 2: p. 17] No security problems are posed when computers process non-sensitive material. Since the merging of the two (computer processing of sensitive material) in first generation computers, computer security has been a concern.

It is generally accepted, that since their initial appearance following World War II, there have been at least three basic generations of electronic computers. The first generation (post WW II-around 1960) consisted largely of "number crunchers" with technology based on relays and vacuum tubes. [Ref. 3: p. 590] These first generation computers lacked operating systems and it usually turned out that the programmer was also the operator of the system. [Ref. 4: p. 2] With the second generation (around 1960-1964) came technology based on solid state components (transistors), batch processing, system operators, and

operating system software in the form of a collection of system routines.

Computer security for the first and second generation computers was not the problem it is today because they were basically one-user-at-a-time systems. In both cases, if sensitive material was processed, the computer facility was physically secured and the computer was run in a "system high" security mode. In the case of the second generation computer jobs classified lower than the system high had their output manually reviewed and downgraded to its proper level of sensitivity.

With IBM's introduction of its 360 system in 1964, the third generation was ushered in with new technology and design philosophies. The technology of the third generation has centered on solid state circuitry (integrated circuits), allowing vast improvements over first and second generation computers in the areas of speed, reliability, capacity, versatility, and cost. In order to meet the growing and varied demand for computers, many manufacturers began to design and market series of general purpose computers. [Ref. 5: pp. 14-16]

Probably the most significant change in computer system design was the implementation of resource sharing. Even before the advent of the third generation, it was apparent that a job running singularly in the computer from start until finish often wasted valuable computer resources

waiting for input/output from the slower peripheral devices, requiring only a small portion of system peripherals, and/or occupying a limited section of core memory. Emerging in such forms as multiprogramming, timesharing, and multiprocessing, resource sharing provided an initial solution to the problem by more efficiently and economically distributing the computer's resources among a group of simultaneous users. [Ref. 6: p. xi]

Some people argue that a fourth generation of computers exists currently. There is no general agreement on what differentiates the fourth generation from the third generation, but some of the proposed characteristics of fourth generation computers are widespread simultaneous interactive computing by multiple users, virtual processing to get the maximum number of user processes into main memory in order to maximize response time to user requests [Ref. 5: pp. 32-33], and Large Scale Integration technology. In general it can be said that the above fourth generation characteristics have expanded the resource sharing capabilities of computers while at the same time lowering the costs. Whether a fourth generation exists or not, there has been an explosion in the usage of resource sharing computers.

The more efficient use of computers through resource sharing has not been without an associated cost. Under resource sharing, the overall complexity of the computer

system increased. The complexity of the security problem also increased. With multiple simultaneous resident user processes, it rapidly became evident that there was a strong need to protect a user from the actions of other users. [Ref. 7: pp. 8-9] This included unintentional actions as well as intentional ones.

Recognizing that a security problem existed was easy, but deciding how to handle the problem was much harder. One possible way to attack the problem is to avoid internal controls through the use of a various computer processing modes. Roger Schell [Ref. 2: p. 25] points out three typical computer processing modes employed to avoid the use of internal controls. In the first mode, each level of classified data is processed on a separate computer dedicated to processing that particular level of classified data. All users of a given computer are cleared for the level of classified processing and are authorized access to all data on the system. This is appealing for real-time and on-line systems, but it can cause duplication or inefficient use of computers. Duplication may exist if more than one classification is processed since a separate computer is needed for each level of classification. Inefficient use may occur if very little processing is done at a given classification level, thus causing underutilization of the computer's resources. A second mode involves scheduling periods of processing so that during any given period, only

one level of classified data is processed. With this approach, all of system memory must be purged between processing. All users during a given period are cleared for the given level of classified processing and are authorized access to all data processed during that period. This approach lacks responsiveness and causes the waste of computer resources during period switching. In the third mode, different classification levels are processed simultaneously. All communication lines are secured and all users are required to be authorized access to all the data. All output from the system is initially classified at the highest level and manually reviewed for downgrading as necessary. This mode is often referred to as "System High". [Ref. 2: p. 25]

Although these approaches enhance the security, they are costly in many ways. There are added financial costs due to increased communication security measures, manual review of output, increased and/or more intensive security clearance investigations (especially for the third approach), and duplicate equipment. Another cost is the increased risk due to expanded exposure of classified data (no way to enforce compartmentalization of data), increased possibility of granting an untrustworthy person a clearance (due to greater number of people requiring clearances), and the error prone tendency of manual review of output for downgrading. Another cost is foregone capabilities such as rapid access

to information stored on the computer, efficient use of the computer, and the establishment of computer networks which serve a diverse and geographically separated user community. [Ref. 2: pp. 25-26] The sum effect of these costs is a reduced level in the qualitative force factors noted earlier.

Even with the methods noted above, there were obvious shortcomings to the noted processing modes. Without internal controls, the security of resource sharing computers can certainly be enhanced, but the resource sharing security issues are not really addressed. Without internal controls, there is no reliable way to protect a given user against the actions of other users. With respect to the handling of classified data, there is the strong need to be able to restrict access to portions of data on a system to a select group of users even though all the users may be cleared for the data. With only the methods noted above, this is an impossibility on resource sharing computers.

In 1967, a special Task Force was organized under the auspices of the Defense Science Board to address the safeguards necessary to adequately protect sensitive information processed on remote access resource sharing computers. In its 1970 report, "Security Controls for Computer Systems", the task force presented a number of policy and technical recommendations aimed at reducing the

threat of compromise of sensitive information processed on such systems. In 1972, the DoD defined a mode which would address the protection problems of remote accessed resource sharing computers. [Ref. 8: p.1] This mode is the multilevel security mode and is defined in the DoD ADP Security manual DoD 5200.28-M as follows:

> A mode of operation under an operating system (supervisor or executive program) which provides a capability permitting various levels and categories or compartments of material to be concurrently stored or processed in an ADP System. In a remotely accessed resource-sharing system, the material can be selectively accessed and manipulated from variously controlled terminals by personnel having different security clearances and access approvals. This mode of operation can accommodate the concurrent processing and storage of (a) two or more levels of classified data, or (b) one or more levels of classified data with unclassified data depending upon the constraints placed on the systems by the Designated Approving Authority (Section V. C., DoD Directive 5200.28). [Ref. 9: p. 11]

In order to fully comprehend its impact upon the security issues of resource sharing computers, one also has to understand the general clearance and access controls as stated in the same manual.

> Personnel who develop, test(debug), maintain, or use programs which are classified or which will be used to access or develop classified material shall have a personnel security clearance and access authorization (need-to-know), as appropriate for the highest classified and most restrictive category of classified material which they will access under system constraints. [Ref. 9: p. 14]

There are several conditions important to the concept of multilevel security. First the system must provide for the concurrent processing of two or more levels of classified

data or one or more levels of classified data with unclassified data. This differs from the dedicated and period oriented computer processing modes noted earlier. Secondly, the system should accommodate access to the computer by personnel having different security clearances and access approvals. This is not covered by any of the mentioned processing modes. Finally, a user must have authorization for the particular data he accesses. In the classified sense, this equates to a need-to-know, but in multilevel secure processing, it has also come to incorporate authorization to perform functions on a computer. As an example, a user must be authorized write access to write to a file. This will be the attitude taken in this thesis. In the broad sense, this is the control aimed at isolating a given user from the actions of the other users.

C. PAST DEVELOPMENTS IN REFINING MULTILEVEL SECURITY REQUIREMENTS

In 1977 the DoD Computer Security Initiative was established under the auspices of the Under Secretary of Defense for Research and Engineering [Ref. 8: p. 1]. The Initiative was aimed at establishing the availability of trusted computer systems. A trusted computer system is "one that employs sufficient hardware and software integrity measures to allow its use for simultaneously processing multiple levels of classified and/or sensitive

information". [Ref. 7: p. 1] In January 1981, the DoD Computer Security Center (CSC) was formed to expand on the work already started by the Initiative.

One of the earliest undertakings of the DoD CSC was the formalization of evaluation criteria through which the DoD could judge the effectiveness of the security employed by its computers. During the formalization, the CSC identified six fundamental requirements that must be met by a computer system in order for that system to be called secure. These are as follow:

> Requirement 1 - SECURITY POLICY - There must be an explicit and well-defined security policy enforced by the system.
>
> Requirement 2 - MARKING - Access control labels must be associated with objects [Passive entities that contain or receive data].
>
> Requirement 3 - IDENTIFICATION - Individual subjects [users, processes, or devices] must be identified.
>
> Requirement 4 - ACCOUNTABILITY - Audit information must be selectively kept and protected so that actions affecting security can be traced to the responsible party.
>
> Requirement 5 - ASSURANCE - The secure computer system must contain hardware/software mechanism that can be independently evaluated to provide sufficient assurance that it enforces the basic requirements.
>
> Requirement 6 - CONTINUOUS PROTECTION - The trusted mechanisms that enforce these basic requirements must be continuously protected against tampering and/or unauthorized changes. [Ref. 8: pp. 3-4]

The CSC points out in its "Trusted Computer System Evaluation Criteria" Final Draft, that these requirements are derived from the need to satisfy basic control

objectives which deal with Security Policy, Accountability, and Assurance. In general a control objective "refers to a statement of intent with respect to control over some aspect of an organization's resources, or processes, or both. In terms of a computer system, control objectives provide a framework for developing a strategy for fulfilling a set of security requirements for a given system." As such, they are a "useful method of formalizing security goals". [Ref. 8: p. 55] In order to understand the implication of the aforementioned requirements, one has to understand the control objectives from which they are derived.

Computer security, in general, is concerned with controlling access to and manipulation of the data processed on the computer. The degree of protection required for a given computer must be based upon the perceived threats, risks, and goals of the owner organization. A security policy is a statement formalizing the requisite protection. More concisely, the Security Policy Control Objective is:

> A statement of intent with regard to control over access to and dissemination of information, to be known as the security policy, must be precisely defined and implemented for each system that is used to process sensitive information. The security policy must accurately reflect the laws, regulation, and general policies from which it is derived. [Ref. 8: p. 55]

Without a stated security policy, there is no measure against which one can assess the degree of security afforded by hardware/software mechanisms.

Both the Security Policy and Marking requirements are derived from the Security Policy control objectives area. The control objectives of the Security Policy requirement may be broken down into two subsets. The first subset of control objectives deal with mandatory security controls.

> Security policies defined for systems that are used to process classified or other specifically categorized sensitive information must include provisions for the enforcement of mandatory access control rules. That is, they must include a set of rules for controlling access based directly on a comparison of the individuals's clearance or authorization for the information and the classification or sensitivity designation of the information being sought, and indirectly on considerations of physical and other environmental factors of control. The mandatory access control rules must accurately reflect the laws, regulations, and general policies from which they are derived. [Ref. 8: p. 56]

Such controls are mandated by established rules that detail how classified or sensitive information is to be handled. Access is restricted in accordance with the clearance/authorization of the user, the classification/sensitivity of the data, and the type of access being attempted.

The second subset of Security Policy control objectives deal with discretionary security policies. The objectives of this subset are:

> Security policies defined for systems that are used to process classified or other sensitive information must include provisions for the enforcement of discretionary access control rules. That is, they must include a consistent set of rules for controlling and limiting access based on identified individuals who have been determined to have a need-to-know for the information. [Ref. 8: pp. 56-57]

Discretionary security differs from mandatory security in that discretionary security is based upon the user specifying the modes of access other users may have to information under his/her control. Discretionary security therefore mediates a users ability to access based on his/her need-to-know that information. As indicated in the previous section, none of the forms of dedicated mode of operation afforded this type of security.

Implicit in a mandatory security policy is the concept that the classification/sensitivity of information should be clearly marked and that such markings should only be alterable by those users who are properly authorized to do so. These goals are clearly outlined in the Marking Control Objectives:

> Systems that are designed to enforce a mandatory security policy must store and preserve the integrity of classification or other sensitivity labels for all information. Labels exported from the system must be accurate representations of the internal sensitivity labels being exported. [Ref. 8: pp. 57-58]

Aside from allowing mandatory security controls to be effective, a side benefit of marking is that all forms of output may be accurately marked.

The Identification and Accountability requirements are derived from the Accountability Control Objectives. These objectives are concerned with individual accountability and are as follows:

> Systems that are used to process or handle classified or other sensitive information must assure individual

> accountability whenever either a mandatory or discretionary security policy is invoked. Furthermore, to assure accountability the capability must exist for an authorized and competent agent to access and evaluate accountability information by a secure means, within a reasonable amount of time, and without undue difficulty. [Ref. 8: pp. 57-59]

Each access to the system and information on the system must be controlled based on who is performing the access and what information they are allowed to access. The identification and authentication of users are, therefore, essential to access control. A related problem on many systems today is a weak accounting system. In many cases, a user may perform a number of functions on a computer with reasonable certainty that there will be no way to determine, after the fact, what he/she did. In order for a system to be deemed secure, it is absolutely necessary that each user be held accountable for his/her actions. Therefore the system must maintain selective accounting information that will allow the proper authorities determine accountability. [Ref. 8: p. 4]

The last two requirements, Assurance and Continuous Protection, are derived from the Assurance Control Objectives. These are:

> Systems that are used to process or handle classified or other sensitive information must be designed to guarantee correct and accurate interpretation of the security policy and must not distort the intent of the policy. Assurance must be provided that correct implementation and operation of the policy exists throughout the systems's life-cycle. [Ref. 8: p. 60]

The mechanisms to accomplish the security policy, marking, identification, and accountability controls are often

embedded in the operating system of the computer. Assurance is necessary to guarantee or provide a degree of confidence that these mechanisms do indeed provide the control that they are intended to and that the mechanisms perform only their intended functions. The general category of assurance can be broken down into two parts. The first, life-cycle assurance, deals with those measures taken by an organization to ensure that the system is designed, developed, and maintained utilizing rigorous and formalized standards and control. During the design, development, and following any changes which may affect the above control mechanisms, the system must be reevaluated to ensure the control objectives are still being met. The second assurance, continuous protection, is concerned with guaranteeing that the security policy is uncircumventably enforced while the system is operating. To this extent, it must be ascertained that there are neither holes through which a user can avoid controls nor avenues that a user can take to alter the control mechanisms. Some of the common measures to accomplish this are isolation of protection mechanism software, testing for the correct operation of operational hardware and software, and hardware and software encapsulation. [Ref. 8: p. 59]

D. FORMAL MODELS

In the last section, we saw that under the assurance of protection requirement, the system must be designed to

guarantee correct and accurate interpretation of the security policy. To show that a design does guarantee such an interpretation is not a trivial process. First. the designers must have a clear definition of the security policy that they are to implement. Although the aforementioned DoD regulations do define the requirements for multilevel security, their English language formulation is not adequate for conclusively demonstrating the correctness and completeness of a security policy implementation. Therefore, most designers rely on formal models to unambiguously describe the security policy being implemented, while at the same time providing a foundation that will allow the implementation to be proved correct and accurate. [Ref. 10: pp. 247-248]

The process of proving the implementation to be correct is known as verification. To do this, the verifier must show the consistency of an implementation with respect to some specification of behavior expected of the implementation. For security, the formal model specifies the behavior expected to be exhibited by the implementation of the security relevant portions of the system. To show consistency between the model and implementation, the verifier often relies on a mathematical proof. [Ref. 11: pp. 1-5] For all but the most minor piece of code, verification can be a long and tedious process even for those well versed in the process. Although verification

is an important aspect of any multilevel secure system, it is beyond the scope of this thesis. Since a formal model will be used to guide the design of the security features of this system, verification may very well be an appropriate follow-on topic for a future thesis.

Several formal models exist, but one of the best known is probably the Bell-LaPadula Model. It has been used as a model for such security related projects as the Kernalized Secure Operating System for the PDP-11, security enhancements to MULTICS for the Air Force Data Services Center, and the SIGMA message system used in the Military Message Experiment. The complete statement of the model is quite lengthy and complex. It can, however, be summed up in two properties as follow:

1. the simple security property: no subject has read access to any object that has a classification greater than the clearance of the subject; and

2. the *-property (pronounced "star property"): no subject has append-access to an object whose security level is not at least the current security level of the subject; no subject has read-write access to an object whose security level is not equal to the current security level of the subject; and no subject has read access to an object whose security level is not at most the current security level of the subject.

In simpler terms, the first property says the user must have a clearance greater than or equal to the classification of the data he/she is attempting to read. The second property has come to be identified with the prohibition of "writing down". In other words, the user can not write a data object to a second data object which has a lower classification

than the first. [Ref. 10: pp. 260-261] This prevents the user from lowering the classification of a given piece of data.

This is gross simplification of the model since it is based on finite state machines and has specific rules for going from one state to the next. It does, however, provide a flavor for the model. It should be mentioned that included in the model are provisions for "trusted subjects". A trusted subject is one which is allowed to operate without being held to the restrictions of the *-property. These subjects can be trusted never to mix data of different classifications. [Ref. 12: p. 65]

Use of this model has uncovered certain problems associated with the model. The model has proved to be overly restrictive with respect to its representation of military security. Although it accommodates the hierarchy of classification, it does not provide for objects which contain multiple levels of classification. A typical message in a military message system is composed of one or more paragraphs, each of which has its own classification. Under the Bell-LaPadula model, the message as an object can only have one classification. In such a situation, the message can be accessed only as a whole with regard to its overall classification. Individual paragraphs would not be accessible individually based on their own classification. [Ref. 10: pp. 262-263]

Except under the trusted subject concept, some of the typical functions of security, such as reclassification, sanitization, and downgrading, are not allowed. The problem here is that there is little guidance on what processes can be trusted. Other problems identified include the possibility of timing channels permitting the exchange of information [Ref. 10: pp. 262-263] and the lack of structure to support application-dependent security rules. As an example of an application-dependent security rule, one observes in a message system, the need to restrict the release operation to those users authorized to do so. [Ref. 12: pp. 66-67]

Due to the special needs for message system security, work is currently underway at the Naval Research Laboratory (NRL) to develop a model which incorporates the application specific security rules of message systems. The development of this model is part of the Navy's Military Message System (MMS) Project which has as one of its goals the development of a family of multilevel secure message systems. Instead presenting the specifics of the model here, an excerpt describing the model can be found in Appendix A.

The MMS model overcomes some of the aforementioned problems and gives security guidance in other areas as well. With the MMS model, the classification of data items may be reduced under certain circumstances, security rules applicable to message processing are incorporated, and

multilevel objects are included. Within the multilevel object concept, the model differentiates between objects (called atoms), which are single level, and containers, which may be multilevel. An atom is the smallest unit of data to which a classification can be attached. The container has a classification of its own and may be made up of atoms (each with its own classification) and/or other containers. An important concept associated with containers is "container clearance required" (CCR). This is an attempt to deal with the aggregation security problem by allowing a minimum clearance requirement for access to the container when necessary.

The model provides for an access set to be associated with each atom and container. The access set consists of a set of pairs, where each pair consists of a user ID and an operation that the respective user can employ on the given atom or container. Provision is also made to define user roles such as releaser, downgrader, and system security officer. It is even possible for a role designator to take the place of a user ID in an access set pair. [Ref. 12: pp. 68-69]

Since this model is tailored to a military message system and the target electronic mail system of this thesis incorporates many of the traits of a military message system, the MMS model is used to guide the design of the security features outlined in this thesis. The model will

be adhered to as closely as possible; however, in some instances deviations may occur. As an example, access sets will be assigned to messages. Messages, however, can be (and more than likely will be) containers since they will contain paragraphs as referable objects. No attempt will be made to assign access sets to these paragraphs. It is the intent of the author to point out all deviations from the model.

III. RELATED WORK

A. INTRODUCTION

In this chapter, we will discuss previous work related to the underlying Integrated Software System(ISS), multilevel security, and security as related to the use of the relational database model. In section B, the development of the underlying features of the ISS is discussed along with the reasons that its electronic mail function was chosen as the basis for this thesis. In section C, the SIGMA message system as a multilevel secure message system is examined. Section D reviews Multisafe for its modular approach to security. Finally, in section E general database management system security mechanisms are discussed.

B. THE INTEGRATED SOFTWARE SYSTEM

Almost anyone who has been working with computer systems for any length of time has experienced delays due to the learning curve associated with each individual application of a given system, the frustration incurred by trying to keep straight in his/her mind the application specific commands as he/she switches back and forth between applications, or the inability to use files created by one application as input to another application. This is due primarily to the lack of integration among applications. As

the use of workstations spreads, the problems associated with the lack of integration becomes intolerable. The aim of the ISS is to provide a degree of uniformity to the workstation environment through the integration of five common applications. The five applications chosen were text processing, form generation, database management, electronic mail, and spread sheet modeling.

Integration is provided by a single conceptual model for the system as a whole and a set of basic commands common to all five applications. This is not to say that there are no application specific operations or commands. What is intended here is a set of five separate applications which share a common intersection of operations, commands, and data structures.

For the ISS, the relational database model is used as the single underlying conceptual model. The underlying data object is the relation. Although a relation may be described mathematically in terms of a subset of the Cartesian product of a list of domains [Ref. 13: p. 19], conceptually it can be viewed as a table. This is how it is viewed and implemented in the ISS. Thus, the table, where the tuples are rows and attributes are column headings, is the primary data object underlying all five applications of the ISS.

With the selection of the relational database model as the single conceptual model and the table as the primary

data object, each application is then viewed as a logical database consisting of a set of tables. Further integration is accomplished by dividing each application's set of tables into three generic subsets or classes: Application Directory Table, Data Table Schema Table, and Data Tables. Each table has key values which allow the unique identification of each row. Any datum in a table can be accessed by specifying the name of the table, the value of the key, and the name of the attribute containing the datum.

The Application Directory Table of a given application contains descriptive and definitional information about the data tables of that application. Each row of the Application Directory Table describes a data table for the given application. A standard schema defines the rows of the directory table, but allows the Application Directory Table to be augmented to accommodate additional data table attributes.

A data table represents the logical file of an application. Much like data elements of modern programming languages, data tables are typed. The type associated with a table is based upon its primary use (i.e., text, form text, database, spread sheet, or mail). Since a primary objective of the ISS is the sharing of data among applications, strong typing is not enforced. Typing is used to categorize data tables in order to logically organize those which are used primarily by the same application.

The Data Table Schema Table defines the structure of the data table, containing a row for each column in the data table. Each row in the Application Directory Table is linked to the Data Table Schema Table and the corresponding data table. The same relationship among the tables exists for each application. Except for the database application, all tables of a given application have the same structure.

Use of a common conceptual model (relational database model) and data object (table) lead to a kernel of operators and their associated operations common to all five applications. Eight primitive operators common to all five applications were defined for the ISS. These operators and their associated operations are as follow:

1. Insert: changes a target table by inserting into it a table at a specified location (ID value) or at the end by default.

2. Modify: alters a target table by changing the values of specified columns in a row or set of rows to new values. Row selection is determined by condition satisfaction.

3. Delete: changes target table by deleting all rows which satisfy a given condition.

4. Project: lists those columns specified, in the order that they were specified.

5. Select: creates a new data table from all rows of a given table which satisfy a given condition.

6. Union: creates a table consisting of the union of the specified tables. If the tables are dissimilar, the prescribed dominant table determines the structure of the resultant table.

7. Sort: creates a new table which has the same structure and data of the specified table, but is sorted on the specified columns.

8. Concatenate: creates a new text data table from any other type of table. The values of specified columns of the operand table are concatenated into a single resultant table, row by row, with each field in a row of the given operand table separated from the next by a space in the corresponding resultant table row.

Taken together, the single conceptual model, the table as the primary data object, and the basic set of common primitive operations present a relatively high degree of integration among the five application areas.

The thesis at hand, however, does not deal per se with ISS as a whole. Rather, it is tailored to developing the conceptual design of those features which would make the electronic mail application of the ISS multilevel secure. Future chapters will deal primarily with the electronic mail application as if it were a stand-alone application.

The question which quickly arises is "Why choose an electronic mail application which is designed as a part of an integrated system?" There are at least four valid reasons for doing so. First, the design for the system exists. By using an electronic mail system with an existing framework, more time can be devoted to the design of the appropriate security features. Second, the conceptual framework of the application exists without the specifics of implementation. The author is therefore not constrained by implementation limitations during the design of the security features. As a result, the security features can become an integral part of the overall conceptual design for the

application, rather than an add-on feature. Third, this particular electronic mail application is based upon the relational database mode. This increases ease of understandability and facilitates certain security features as will be described in later chapters. Finally, although it is not a major factor of this thesis, it is hoped that the security features designed for the electronic mail application will be general enough to allow the incorporation of one or more of the other applications, thus broadening the usefulness of the system. In any case, this last point is left for future study.

C. SIGMA AS A SECURE MESSAGE SYSTEM

Since 1960 much of the military message processing has been automated in what appears to be three stages. The first stage of automation emerged during the 1960's with the advent of communication networks, such as the Automatic Digital Network (AUTODIN), for transferring formal messages between military organizations. In the early 1970's, the second stage saw the introduction of telecommunication center message systems such as the Local Digital Message Exchange (LDMX). The purpose of these systems was to automate some of the message processing tasks. Error checking and statistics gathering are examples of the types of tasks automated by the second stage systems. The final and current stage was started recently with systems like the

National Military Intelligence Center Support System (NMIC-SS) and SIGMA. These systems are characterized as user-oriented message systems because they provide direct aid to the drafters and recipients of messages [Ref. 14: pp. 1648-1649].

In terms of security, many of the DoD message systems are designed to operate in the System High Mode. Few, however, have been designed to operate in a Multilevel Security Mode. One such system, though, is SIGMA. [Ref. 15: p. 3]

SIGMA was developed as an experimental system in conjunction with a joint experiment (Military Message Experiment) by the Navy, Defense Advanced Research Projects Agency (ARPA) and CINPAC to demonstrate and assess the utility of an interactive message handling system to operational military users. During the experiment SIGMA was used by approximately 100 officers in the Operations Directorate and the command center of the Commander-in-Chief, Pacific (CINPAC). It was connected to an LDMX, allowing the users to send and receive formal messages over AUTODIN. SIGMA also supported informal messages and a class of messages know as formal memoranda. This latter class was composed of on-the-record messages between the SIGMA users.

Many features were incorporated into SIGMA in order to make it highly useful for military applications. Like many

other interactive message systems, it supported the delivery and display of incoming messages, composition and transmission of outgoing messages, and storage and retrieval of messages. SIGMA also provided computer aided distribution of messages. In order to accommodate the desire of the CINPAC Operations Directorate not to have full automatic distribution, SIGMA presented all messages for the Directorate to a special user who reviewed the message and determined the appropriate distribution within the Directorate. Some of the other useful features included on-line action logs and readboards, computer-based message coordination, automation of the release function, message archival, and message retrieval from archival storage.

Even though SIGMA was implemented on a non-secure operating system (TENEX), its user interface was designed as if it were running on a multilevel secure kernel. This was done so that the interface would remain unchanged if SIGMA were ever implemented on a secure operating system. SIGMA's secure interface includes a multilevel user terminal. Some of the features of the terminal include the division of the terminal screen into windows, each acting as a logically independent terminal; two sets of security lights to indicate respectively the highest classification being displayed on the terminal and the classification of the window where the cursor is located; and special function keys for security relevant operations. [Ref. 14: p. 1650]

It was envisioned that the security kernel which the interface would run on would implement the Bell-LaPadula security model. To this extent, a trusted process facility was included in the interface. It was discovered, however, that requiring SIGMA to enforce the Bell-LaPadula model presented several problems.

First, military message systems must be able to operate on multilevel objects such as messages and message files. As noted in Chapter 2, the Bell-LaPadula model does not support multilevel objects. Second, while downgrading is a common operation required by message system users, it is prohibited by the Bell-LaPadula model. Third, message system security requires certain application specific rules (e.g., message release), yet the Bell-LaPadula model has no provision for handling these rules.

Without a multilevel object capability, SIGMA users are forced to perform a downgrading operation where one should not be required. An example of this occurs when a user extracts a confidential paragraph from a secret document. Since the Bell-LaPadula model requires the secret classification to be carried forward with the extracted paragraph, the user faces working with an incorrectly classified paragraph or invoking the downgrading procedure. To do this the user must copy the paragraph to a new document with a classification of secret and then invoke the downgrade operation on the new document to lower the

classification to confidential. Downgrading, the second problem, is performed by a trusted process which can, under certain circumstances, violate the Bell-LaPadula model. To handle SIGMA's third problem, software was developed to perform the required checks. This software, however, is external to both the kernel and the trusted process. [Ref. 14: pp. 3-4]

It is in light of such problems and their related patches that the Military Message System's security model is being developed. [Ref. 15: p. 1] It is tailored for the needs of a message system and, therefore, designed to overcome the problems noted here. For this reason, it has been chosen to guide the development of the security measures presented in this thesis.

D. MULTISAFE: A MODULAR APPROACH TO SECURITY

In the development of most software systems, the modularization of the system is of major issue. When developing a secure software system, the issue of modularization becomes extremely important. The reason for this evolves from the assurance and continuous protection requirements discussed in Chapter 2.

The basic underlying concept of these requirements is that there must be some way to guarantee that the security mechanisms do indeed provide the protection dictated by the given security policy; that they perform only their intended

functions; and that the mechanisms provide uncircumventable, continuous protection. For secure systems, such a guarantee usually means verification of the implementation with respect to a security model as noted in Chapter 2. Even under the best of conditions, verification is a long, tedious, and difficult process. If, however, the security mechanisms are distributed throughout the system, verification becomes virtually impossible. The design, therefore, must provide the most favorable conditions for establishing the above guarantee.

One of the current philosophies oriented to establishing favorable conditions for verification centers around the concept of encapsulation. Under the concept of encapsulation, a section of a given system is circumscribed. Access to the circumscribed section can only be made via prescribed paths and all accesses are controlled.

The mechanism providing the encapsulation is often viewed as a reference validation monitor because its job is to mediate all references (accesses) to the circumscribed section. Such a reference validation mechanism has three basic properties.

1. It must be tamperproof.
2. It must always be invoked.
3. It must be small enough to be subjected to analysis and tests, the completion of which can be assured. [Ref. 12: p. 71]

In essence, the reference validation mechanisms are also circumscribed.

Needless to say, modularization plays a key role in the circumscription of the desired section and the development of the reference validation mechanism. All the functions of the circumscribed section must be defined, isolated, and incorporated into the overall module to be circumscribed. All interfaces to the module must be clearly defined and it must be assured that no alternative paths (interfaces) into the module exist.

A similar process occurs with the reference validation mechanism. All of its functions must be clearly defined, isolated, and incorporated into the reference validation module. As mentioned earlier, the role of the reference validation mechanism is to mediate accesses to the circumscribed module. It, therefore, logically sits between the users and the circumscribed section, filtering the users' accesses to the circumscribed section. Since the reference validation mechanisms, in essence, will provide the multilevel security of the system through its mediation of accesses, it will have to undergo verification. It should, therefore, be kept as small as possible, and only necessary functions should be incorporated in it.

This is the type of approach taken by the MULTISAFE system. MULTISAFE is a MULTImodule system for supporting Secure Authorization with Full Enforcement for database

management. As the break-out of its name would indicate, it is designed to provide securely controlled database access and it is the claim of its authors that it is designed:

1. to be verifiably secure,
2. not to incur a prohibitive performance penalty,
3. to produce a modular system in accordance with the structured approach to design,
4. to be naturally extendible to the protection of distributed data, and
5. to provide mechanisms flexible enough to adhere to complex protection policies. [Ref. 16: p. 382]

The MULTISAFE design centers around the division of the data management system into three functionally separate modules: the user application module (UAM), the data storage and retrieval module (SRM), and the protection and security module (PSM). All three modules are designed to function in a concurrent fashion and are treated as separate and isolated processes. Although the modules are logically separate, the modules may or may not be physically separate. Physical separation is not critical to the security of the system but does enhance performance due to actual concurrency of operations. [Ref. 16: pp. 384-385]

With the separation of functions, the role of the PSM is to perform only security checks (reference validation). As a separate module, the PSM offers fine granularity and its sophistication may vary to accommodate complex protection policies. Three classes of access decisions are supported:

data-independent, data-definition-dependent, and data-value-dependent. Examples of data-independent access control conditions are user and/or terminal identification, time of day session initiated, and system status. Data-definition-dependent conditions limit access based on attributes (relation and attribute names) but not data values. Data-value-dependent access controls are a function of the values of attributes. As designed, the PSM performs only discretionary access control. It, therefore, does not enforce the mandatory access control necessary for multilevel security. [Ref. 16: p. 385]

The UAM provides the interface between the user and the system. It reads and analyzes user queries and formats and displays results. Many of the functions traditionally located in the operating system are executed within the UAM. In a multiuser environment, the UAM may be viewed in a number of ways. One view is as a conventional multiprogrammed processor which has disjoint user address spaces. An alternate method has at least part of the UAM residing in each of the users' terminals. In the case of intelligent terminals, each user's software and local data buffers are physically separated from the other users.

The SRM resides on a separate processor. Its primary jobs include database storage management and the performance of database accesses for the PSM and UAM. Since it resides on a separate processor, the SRM can compute such values as

SUM, COUNT, and AVERAGE and perform special functions such as JOIN, PROJECTION, and the establishment of views for a relational database. In addition to its database functions, the SRM maintains private application files and handles the simple input/output operations for these files. [Ref. 16: p. 386]

As noted earlier, all interfaces to a circumscribed section must be clearly defined. MULTISAFE has attacked this problem by viewing the communications as going between an unsensitive part (UAM) and a sensitive part (SRM) with a gate (PSM) between them. With this in mind, it was decided that all communications between modules would be handled via messages. In order to assure all requests for SRM information are mediated, the PSM controls the intermodule communications. A general scenario of how a request is fulfilled is as follows:

1. The UAM polls the terminals for user requests.

2. Once a user request is received, the UAM formats the request into message format, tags the message with a unique terminal identifier identifying the source, and places the message in the UAM's memory.

3. When the PSM is ready to process a new request, the PSM notifies the UAM. Upon acknowledgement of a request pending, the PSM retrieves the request from the UAM's memory and writes it to the SRM's memory.

4. The PSM retrieves the appropriate authorization check information from its database and starts the checking process. If additional information is needed from the user, the PSM sends a message to the UAM.

5. The SRM performs the retrieval and specified data manipulation.

6. When the SRM is ready to send a block of data, it notifies the PSM.

7. The PSM retrieves the data from the SRM's memory.

8. The PSM performs data dependent checks on the data.

9. If the access is authorized, then the PSM writes the data to the UAM's memory and notifies the UAM.

10. The UAM then sends the result to the requesting user. [Ref. 16: pp. 399-400]

The security provided by MULTISAFE is based upon several assumptions and definitions. The assumptions made are as follow:

1. Physical access is controlled. Access to the system is through terminals only.

2. PSM programming is impervious to modification. All of the PSM software is implemented via ROM.

3. User identification is assumed to be correct.

4. Users are separated in the UAM. Primary memory protection is provided by the UAM to prevent a user from interfering with another user's processes, data, and/or messages.

5. Security is limited in scope. Security is limited to access controls and, therefore, does not incorporate information flow controls or inference controls.

6. Only discretionary access controls are enforced.

With these underlying assumptions access and data security can be defined. Access includes all operations which read, write, or store data on the system. It also includes all operations which set, alter, or display authorizations. The data of a system is secure if the enforcement process only allows those access authorizations specified by the authorizer. MULTISAFE's data security may alternately be

stated in terms of four conditions. First, all authorizations are properly stored in the PSM database. Second, the PSM's access decisions are correct with respect to the access request, authorization information stored in the PSM's database, and the state of the system. Third, all requests for access are mediated by the PSM. Fourth, data may move between the user and the database only as a response to an authorized access request. [Ref. 16: pp. 387-388]

Although the MULTISAFE system does not enforce mandatory security access controls, it does provide a good approach to modularization of a secure system. Much of the modularization approach will be used in the design of the multilevel secure electronic mail application proposed in this thesis. Modifications will be made to accommodate mandatory access controls and other features incorporated by this thesis.

E. GENERAL DATABASE MANAGEMENT SYSTEM SECURITY MECHANISMS

Although multilevel security requirements exist within the DoD, the security requirements of the typical database management system (DBMS) users are generally less stringent. Consequently there appear to be no off-the-shelf multilevel secure systems available. This is not to say that there are no security mechanisms incorporated in the off-the-shelf DBMS's or no ongoing research into security of DBMS's.

The concerns of the typical DBMS user center around two types of database protection: integrity preservation and security (access control). Integrity preservation is oriented to preventing incorrect data from entering the data base as a result of nonmalicious errors such as mistyping or programming errors. The thesis at hand, however, is interested in the access control aspect of database protection. Therefore, a brief examination will be made of some of the mechanisms and means available to DBMS users to control access. It should be noted that not all DBMS's incorporate the mechanisms which are to be examined.

Some DBMS's have the means to enforce varying degrees of discretionary access control. One method, exhibited by System R and Query-by-Example (QBE) is the maintenance of table-of-rights for users. A table-of-rights operates on the same basic principle as the access sets of the MMS security model described in Chapter 2. Such a table specifies the capabilities of the listed users over given information in the database. [Ref. 13: p. 355]

Views may be used to limit access to a portion of the database. The role of a view is to define a portion of the conceptual database. In much the same way that a schema defines the makeup of the conceptual database, a subschema traditionally defines the view. Such a subschema includes only those attributes of relations found in the desired portion of the conceptual database. To the user controlled

by a view, it appears as if the database is comprised only of those parts defined by the view. Access control is obtained by not incorporating attribute information of protected values, thus not making them referable. [Ref. 17: p. 226] A number of systems such as System R, QBE, and IMS incorporate some sort of view mechanisms. [Ref. 13: p. 356]

Another means of controlling access to database information is query modification. In query modification, the system modifies queries with extra conditions which must be met. The extra conditions filter out the sensitive information. It should be noted that under both views and query modification, some inefficiencies exist because information is retrieved which is not passed on to the users. [Ref. 17: pp. 226-227]

By definition, a view defines a portion of the conceptual database. In essence, query modification also defines a portion of the conceptual database and is therefore a means of establishing a view. The difference between the view in the traditional sense and query modification lies in the way the two are commonly implemented. As noted, the view is generally established through a subschema which acts as a template for what the user can see. Unless the subschema is equivalent to the schema for the whole conceptual database, the user typically

sees only a subset of all the attributes of the whole conceptual database.

Query modification can accomplish the same thing by modifying a user's query to eliminate the attributes not maintained in the view. The subtle difference is that in query modification, the user can see the entire makeup of the conceptual database, but is not allowed to retrieve information from those attributes eliminated by the query modification. Query modification does, however, has a dynamic quality which lends itself to the task of multilevel security. This dynamic quality is the elimination of all values of given attribute which do not meet a predefined condition. In the case of multilevel security, it is the elimination of all information for which the classification is not less than or equal to the clearance of the requesting user.

In essence, mandatory access will be based on a universal view mechanism built into the software. This mechanism will be based on the concept of query modification and as described above will filter out all information from a query which is classified higher than the clearance of the requesting user.

IV. TABLES AND ATTRIBUTES

A. INTRODUCTION

The tables and attributes necessary to support a multilevel secure stand alone version of the ISS mail application are defined in this chapter. Section B presents an overview of the system of tables used by the mail application. Section C discusses the Schema Table and the role it plays in defining the composition of other tables. The maintenance of user specific security information in the Users Table and the need for bitmap translation tables is shown in Section D. Section E describes the Mail Directory Table and section F presents the Mail Data Table along with the Body Table. Finally, the makeup and various aspects of four security specific table types (Directory Table Access Table, Mail Message Control Table, Access Control Table, and Object Control Table) are covered in sections G through J.

It should be noted that the non-security related aspects of the tables presented here are taken from Harrison and Thompson's thesis [Ref. 1] on the Integrated Software System. Only very minor modifications are being made to these aspects and will be noted when made. As noted earlier, individual references to each item taken from their thesis would be cumbersome and a general acknowledgement is made instead. The security features presented here,

[illegible]

however, are the subject of this thesis. An initial presentation of the features were presented in Harrison and Thompson's thesis in preparation for this follow-on thesis. Since that presentation, some security features have been changed, therefore causing discrepancies between the former and current theses. With regard to the security features, the current thesis should be the appropriate point of reference.

B. OVERVIEW

Twelve table types are discussed in this chapter. Two are used to hold the classified mail messages and the remaining ten are used to enforce the access mediation required by the Military Message Systems (MMS) security model. Figure. 4.1 shows all twelve tables and the linkages among them. A brief explanation of each is offered below:

1. Schema Table: provides information of the structure of the tables that the system is working with.

2. Users Table: used during logon to authenticate user and establish access clearance level.

3. Trans_role Table: used to translate user readable role description into machine readable role description. Like the Trans_compart and Trans_caveat Tables, the Trans_role Table facilitates user friendliness. The user inputs actual or mnemonic representation of roles and the system obtains a bit-map representation from the Trans_role Table.

4. Trans_compart Table: used to translate user readable compartment description into a machine readable description. Promotes user friendliness. See Trans_role Table.

5. Trans_caveat Table: used to translate user readable caveat description into a machine readable description. Promotes user friendliness. See Trans_role Table.

6. Mail Data Table: holds the header information of mail messages and pointers to the associated Body Tables which hold the text of the mail messages. There is a separate Mail Data Table for each addressee in the mail system. An addressee is an entity, such as a user or project, which is authorized to receive mail.

7. Body Table: used to hold the text of a single mail message. The text and its associated header information form one mail message.

8. Mail Directory Table: used as a central directory to locate individual Mail Data Tables. Contains pointers to the Directory Table Access, Mail Message Control, and Mail Data Tables. Provides the first layer of control by providing minimum clearance, compartment, and caveat requirements that a user must meet to perform any access operation on the associated Mail Data Table (to include sending mail to the addressee represented by the Mail Data Table).

9. Directory Table Access Table: used to provide second layer of access mediation by identifying those users who are allowed access to the associated Mail Data Table. There is one Directory Table Access Table for each Mail Data Table on the system. Unless in the access list, a given user will not be able to perform any access operation on the associated Mail Data Table (to include sending mail to the addressee represented by the Mail Data Table).

10. Mail Message Control Table: provides the third layer of access mediation by providing the minimum clearance, compartment, and caveat requirements that a user must meet in order to perform any access operation on the associated mail message. There is a row for each mail message in the Mail Data Table. It also contains pointers to the Access Control and Object Control Tables.

11. Access Control Table: provides the fourth layer access mediation by identifying those users allowed access to the associated mail message and the access operations that they are allowed to perform.

12. Object Control Table: provides fifth layer of access control by allowing mandatory access control over all

objects in the associated mail message. It holds the classification, compartments, and caveats associated with each object in the mail message.

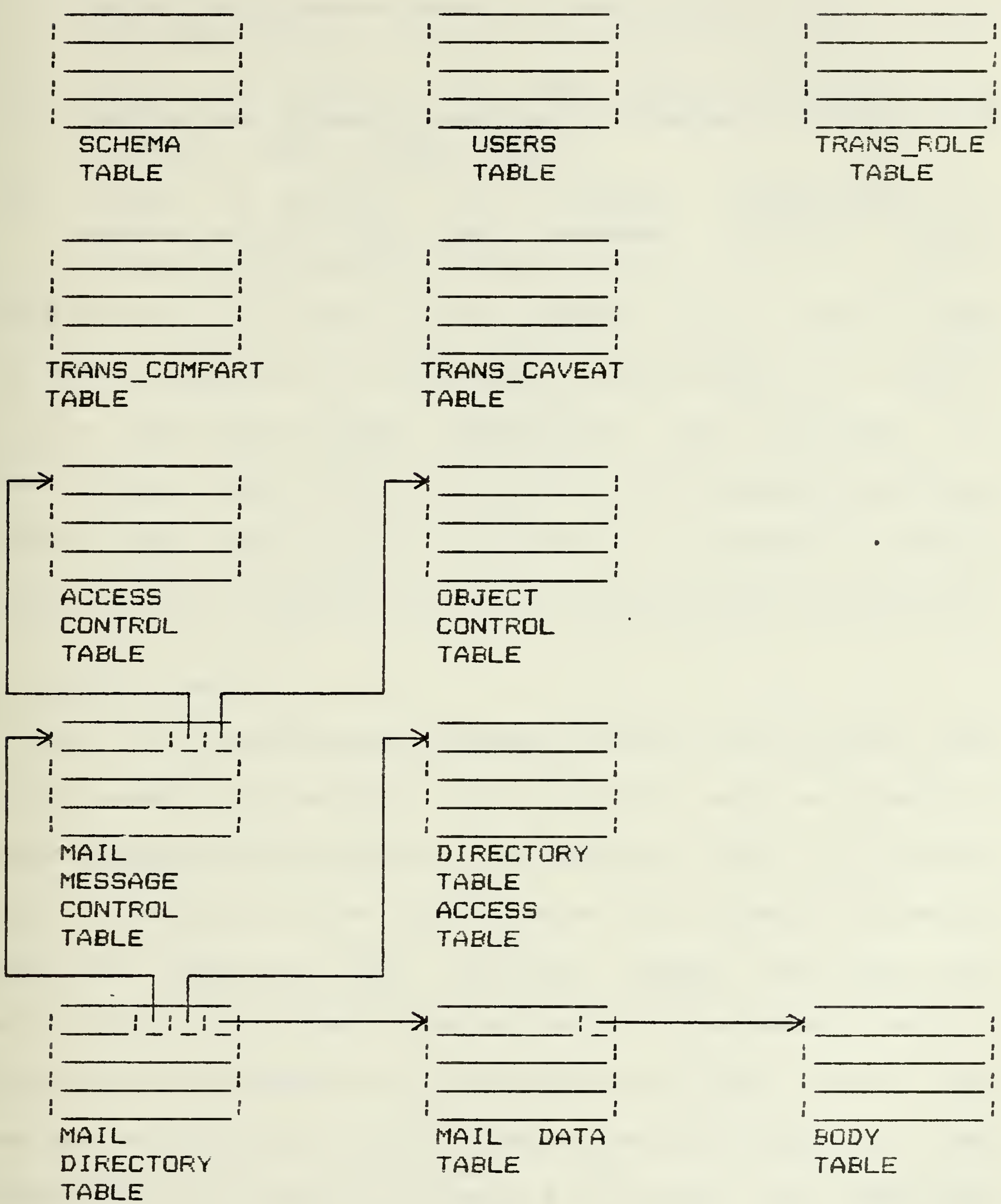

Figure 4.1. Mail Application Table Types

As is indicated, access mediation is approached in a layered fashion:

1. Authorized system usage is determined at logon using the Users Table.

2. Access to general Mail Data Table information is determined though the Mail Directory Table and Directory Table Access Table.

3. Access to specific mail message information is controlled by the Mail Message Control, Access Control, and Object Control Tables.

The details of the tables are presented below. Their usage for access mediation is discussed in more detail in chapter 5. Only three basic access operations are envisioned for this system: Read, Write, and Send. The functions performed during these operations are discussed in chapter 5. Chapter 6 discusses how the user might use these access operations.

C. SCHEMA TABLE

As in the Harrison and Thompson thesis, the Schema Table is discussed first so that its structure may be used to describe subsequent tables. The Schema Table is a single table containing one row (tuple) for each different attribute found in the various tables of the mail application. In general, the purpose of the Schema Table is to provide information on the structure of the tables that the system is working with. The columns (attribute) of the Schema Table are shown in Figure 4.2 and are self-described in Figure 4.3 by rows (tuples) from the Schema Table itself. It should be noted that in the mail application

presented here, all tables follow a standard format. The Schema Table and some other tables are discussed here not so much for their usefulness to the subject of this thesis, i.e. multilevel security, but for ease of cross reference between the current and former thesis and the ease of possible future incorporation of other ISS applications.

The ID column is a six digit field representing the display order of the rows as a table and the conceptual ordering of the rows in the database. This does not mean, however, that the actual physical implementation of the application must store the relation in this fashion. As a whole, the attributes established in this chapter are not meant to be fixed for implementation as described. They are described in terms of fixed representations in order to establish a foundation for the conceptual design and may take entirely different forms when implemented. They should, however, perform the same functions as indicated in this thesis.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE

Figure 4.2. Schema Table Schema

The ID column is incorporated into all tables in the application and corresponds to the record numbers found in such systems as DBMS II. In order to avoid redundancy, the

ID column will not be redescribed in the description of subsequent tables. It should be noted that since the column names of a table represent the attributes of relations and the rows of a table represent the tuples of a relation, the paired terms will be used interchangeably in follow-on descriptions.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	ID	INTEGER	6	OFFSET	-ALL
	NAME	CHAR	20		SCHEMA
	TYPE	CHAR	8		SCHEMA
	WIDTH	INTEGER	8		SCHEMA
	SYNONYM	CHAR	0		SCHEMA
	TABLE	CHAR	0		SCHEMA

Figure 4.3. Self-describing Tuples of Schema Table

The NAME column is the textual name of the attribute which appears in one or more tables of the application. The physical data type and maximum size of data found in the "NAME" column is described by TYPE and WIDTH columns. As a convention "0" in the width column means "of varying length". The SYNONYM column holds the names of other columns in the system which have the same characteristics (TYPE and WIDTH) and may hold data compatible with the column being described. TABLE contains the name of a particular table or type of table where the column being

described can be found. By convention a simple literal indicates a particular table, a "-" followed by a literal indicates all of a particular table type or class.

Since all the tables in the application follow fixed formats, the Schema Table can be created entirely at system generation. In Figure 4.3 we see those Schema Table tuples which define the attributes of the Schema Table itself. As an example of how the information of the Schema Table is broken down we look at the first tuple which describes the ID column. In this tuple we see that the ID column holds a six digit integer value. The column OFFSET has the same characteristics as the ID column. The ID column appears in all of the tables found in the Application.

D. USERS TABLE AND BIT MAP TRANSLATION TABLES

The Users Table is used by the system to mediate initial access to the system and establish the clearance level of each user after he/she has logged onto the system. A separate row is maintained for each authorized user of the system. Since access mediation also involves the clearance level of the terminal being used, this information can also be stored in the Users Table. Access to the table is limited to a user operating under the System Security Officer (SSO) role (see Appendix A). The Users Table is shown in Figure 4.4 and the attributes are described by their corresponding Schema Table tuples in Figure 4.5.

In the case where a user is being identified, USER_NAME is the actual name of the user described by the tuples. USER_ID is a unique alphanumeric string which the system uses to represent the user. Since USER_ID is unique for each individual user, it is not necessary that the user's actual name be manipulated to make it unique. For terminal identification, USER_NAME and USER_ID may be used to identify the terminal and/or computer port. AUTHENTIC is the authentication information that must be supplied by the user during log on in order to be granted access. AUTHENTIC is not applicable to terminals. For purposes of illustration only, it is assumed that a one-way encrypted password is stored in the AUTHENTIC column. As with any of the other fields, AUTHENTIC is only meant to conceptually represent required information and is not meant to place restrictions on actual implementation. In an actual implementation, AUTHENTIC could be a pointer to a table containing information required for a voice print verification of the user.

ID	USER_NAME	USER_ID	AUTHENTIC	CLEAR

COMPART	CAVEAT	ROLE

Figure 4.4. Users Table schema

CLEAR indicates the highest classification of data that the user or terminal may receive. For the application at hand, access mediation will be based on the maximum common clearance level (to include clearance as well as compartments and caveats) between the user and the terminal he/she is operating from. Thus, the user will be granted access to data classified less than or equal to the classification indicated by the minimum of the user and terminal CLEAR values. The standard DoD classification hierarchy of UNCLASSIFIED, CONFIDENTIAL, SECRET, and TOP SECRET is assumed and, for the purposes of this thesis, are represented as U, C, S, and T. COMPART and CAVEAT indicate respectively the compartments and caveats the user or terminal is allowed to operate under. The common set of compartments will be the intersection of the user's and terminal's COMPART values and the common set of caveats will be the intersection of the two CAVEAT values. It is envisioned that COMPART and CAVEAT will be implemented via bit maps where each "1" bit represents access under the particular compartment or caveat represented by that position. ROLE is also envisioned as a bit map and indicates the roles the user can assume. Role is not applicable to terminals. Bit maps have been chosen to represent compartments, caveats, and roles because of the possibility of multiple values in each case. For example, a user may be authorized to operate under downgrader and SSO

roles. Bit maps allow easy representations of multiple values.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	USER_NAME	CHAR	20		USERS
	USER_ID	CHAR	10	OWNER USER	USERS
	AUTHENTIC	CHAR	12		USERS
	CLEAR	CHAR	1	M_CLEAR CLASS O_CLEAR	USERS
	COMPART	BOOLEAN	16	M_COMPART O_COMPART	USERS
	CAVEAT	BOOLEAN	16	M_CAVEAT O_CAVEAT	USERS
	ROLE	BOOLEAN	16		USERS

Figure 4.5. Schema Table rows for Users Table

The use of bit maps, however, does imply the need for translation from a user readable representation to the bit map and possibly vice versa. As might be expected, tables are used for the translation. Figure 4.6 shows the common schema used for the Trans_role, Trans_compart, and Trans_caveat Tables. The corresponding Schema Table rows are given in Figure 4.7. USER_FORM is the user readable representation which corresponds to SYS_FORM, the system readable representation.

ID	USER_FORM	SYS_FORM

Figure 4.6. Trans_role, Trans_compart, and Trans_caveat Tables' schema

Such tables would be created at system generation with unalterable ID and SYS_FORMAT columns. The ID column is numbered one-up to the number of bit positions in the bit map. The SYS_FORMAT is all zeroes except in the bit position corresponding to the associated ID value. Bit positions are numbered one-up from right to left. The USER_FORM may be modified by a user in the SSO role to establish a user readable representation which will from then on correspond to the SYS_FORM value. Therefore, when a user readable representation is detected on input, the system can establish the proper system readable representation through the translation table. Once translated, boolean operators could be used to derive multiple values and for access checks.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	USER_FORM	CHAR	15		TRANS_ROLE TRANS_COMPART TRANS_CAVEAT
	SYS_FORM	BOOLEAN	16		TRANS_ROLE TRANS_COMPART TRANS_CAVEAT

Figure 4.7. Schema Table rows for Trans_role, Trans_compart, and Trans_caveat Tables

As indicated, three such tables, Trans_role, Trans_compart, and Trans_caveat, would be needed to translate roles, compartments, and caveats respectively. Figure 4.8 shows how a portion of the Trans_role Table might appear conceptually.

1	RELEASER	0000000000000001
2	DOWNGRADER	0000000000000010
3	SSO	0000000000000100

Figure 4.8. Trans_role Table example

E. MAIL DIRECTORY TABLE

The role of the Mail Directory Table is to describe the Mail Data Tables. The Mail Directory Table has a row for each logical Mail Data Table on the system and is used by the system and the user to locate particular Mail Data Tables. Figure 4.9 shows the attributes of the directory table and Figure 4.10 presents the Schema Table rows defining the Mail Directory Table attributes.

TABLE_NAME is a unique name for the Mail Data Table described by the tuple and acts as a pointer to that Mail Data Table. Since TABLE_NAME is the name for the Mail Data Table of a particular addressee, it reflects the addressing scheme used to deliver mail messages. For the purpose of this application, an address is assumed to be an assigned

unique name associated with an addressee. For example, mail might be sent to the author by using the address RWYATT. RWYATT would then be incorporated into the name of the Mail Data Table for the author. A possible name might be Mail_Data_RWYATT. COLUMNS lists the column names found in the Mail Data Table. COLUMNS and the Schema Table are used to completely describe the associated Mail Data Table. The names of those fields which comprise the key for the Mail Data Table are held in the KEY field. For the purposes of this application, both the the date-time-group (DTG) and ID fields will be used as the key fields. In particular, DTG will be used for specific retrieval from storage. This deviates somewhat from the Harrison and Thompson thesis which states that ID will be the key field.

ID	TABLE_NAME	COLUMNS	KEYS	VIRTUAL	CONDITION	GLOBALS	OWNER	DESCRIPTION	M_CLEAR	M_COMPART	M_CAVEAT	DTAT	MMCT

Figure 4.9. Mail Directory Table schema

VIRTUAL is a logical field indicating whether the Mail Data Table is to be composed from other Mail Data Tables, and if true, then CONDITION is a list of the tables that the

virtual table is to be composed from. In such a case, the corresponding Mail Data Table does not exist on a permanent basis. Instead it is created from other Mail Data Tables upon request and granting of access to the virtual Mail Data Table.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	TABLE_NAME	CHAR	20		DIRECTORY
	COLUMNS	CHAR	0		DIRECTORY
	KEYS	CHAR	8		DIRECTORY
	VIRTUAL	BOOLEAN	1		DIRECTORY
	CONDITION	CHAR	0		DIRECTORY
	GLOBALS	CHAR	0		DIRECTORY
	OWNER	CHAR	10	USERID USER	DIRECTORY
	DESCRIPTION	CHAR	0		DIRECTORY
	M_CLEAR	CHAR	1	CLEAR CLASS O_CLEAR	DIRECTORY MSG_CON_TBL
	M_COMPART	BOOLEAN	16	COMPART O_COMPART	DIRECTORY MSG_CON_TBL
	M_CAVEAT	BOOLEAN	16	CAVEAT O_CAVEAT	DIRECTORY MSG_CON_TBL
	DTAT	CHAR	20		DIRECTORY
	MMCT	CHAR	20		DIRECTORY

Figure 4.10. Schema Table rows for Mail Directory Table

The use of CONDITION here differs from that proposed by Harrison and Thompson. In their thesis, CONDITION held the

series of operations necessary to generate the desired virtual table. Due to the proposed modularization, this has been changed to indicate the source Mail Data Tables only. The result is that upon request of access to the virtual table, the system will compose a temporary Mail Data Table of those mail messages from the indicated source tables for which the requesting user is authorized access.

GLOBALS is a text string which may contain data related to formatting, display mode, or other parameters useful to the system. OWNER is the userid of the designated administrator of the associated Mail Data Table. DESCRIPTION is a short narrative description of the Mail Data Table.

M_CLEAR, M_COMPART, and M_CAVEAT correspond to the minimum clearance, minimum compartments, and minimum caveats that a user must meet in order to read the corresponding Mail Directory tuple and have access to the associated Mail Data Table. A blank field for the M_CLEAR column and/or all zeroes in M_COMPART or M_CAVEAT columns means there are no minimum requirements for the corresponding category. DTAT and MMCT hold unique names which point respectively to the Data Table Access Table and Mail Message Control Table. The Directory Table Access Table is an access list which specifies those users authorized read access to the corresponding Mail Directory tuple and access to the associated Mail Data Table. The Mail Message Control Table is used to enforce mandatory and discretionary control over

individual mail messages found in the associated Mail Data Table. The Directory Table Access Table and the Mail Message Control Table are described in more detail in subsequent sections of this chapter.

The tuples in the Mail Directory Table are created as a coordinated effort between the System Administrator and the System Security Officer in response to a user request for a new Mail Data Table. The System Administrator collects the required information from the user. After all the information has been gathered, he/she allocates space for the new Mail Data Table and turns the information over to the SSO. Under the SSO role, a new tuple is created in the Mail Directory Table. The non-security related fields are filled in from the information provided by the System Administrator. The DTAT and MMCT fields are filled in with the corresponding table names. If the user requesting a Mail Data Table desires, the M_CLEAR, M_COMPART, and M_CAVEAT fields will be filled in as directed. Otherwise they are filled in to indicate no minimum requirements. The default of no minimum requirements is appropriate since a newly created Mail Data Table would contain no classified information. The indicated owner (or SSO role) may change the minimum requirement fields at any time after the corresponding Mail Data Table has been created. The other fields of the tuple can be modified only under the SSO role.

[illegible]

F. MAIL DATA TABLE

The Mail Data Table is the actual repository of mail messages. Each row in the Mail Data Table represents a separate mail message. The schema of the Mail Data Table is shown in Figure 4.11. Figure 4.12 presents the Schema Table tuples which describe the Mail Data Tables.

ID	VIEWED	FROM	TO	COPY_TO	DTG	SUBJECT	BODY

Figure 4.11. Mail Data Table schema

The VIEWED attribute is a boolean value indicating whether or not the message has been read by the administrator of the mail data table. The header of the mail message is made up of the FROM, TO, COPY_TO, DTG, and SUBJECT attributes. FROM is filled in by the user but must be a valid address for delivering mail to a Mail Data Table for which the user is the designated administrator (OWNER). TO and COPY_TO are also filled in by the user and must be valid addresses. SUBJECT is filled in by the user, but no restrictions are placed on its contents.

The DTG (date time group) will be a system time stamp which initially indicates when the message (tuple) was created via a write operation to the Mail Data Table or when

the message was processed by a send instruction. The DTG of a message processed by a send instruction is always overwritten with the date-time-group at the time of the send operation. The DTG of a send operation has a suffix of "S". If the DTG is empty preceding a write operation, the DTG will be filled in with the date-time-group associated with the writing of the mail message tuple. In this case the DTG will have a suffix of W. If the DTG is non-empty prior to the write, the DTG will remain as it is. Thus the user may preserve the previous DTG value or cause the generation of a new one by blanking out the DTG field prior to writing. The assignment of the DTG as described above maintains the uniqueness which it requires as a key attribute.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	VIEWED	BOOLEAN	1		MAIL
	FROM	CHAR	0		MAIL
	TO	CHAR	0		MAIL
	COPY_TO	CHAR	0		MAIL
	DTG	CHAR	20		MAIL
	SUBJECT	CHAR	0		MAIL
	BODY	CHAR	20		MAIL

Figure 4.12. Schema Table rows for Mail Data Table

The attribute BODY holds the name of the Body Table containing the text of the mail message. This is a slight

deviation from that proposed by Harrison and Thomoson. They proposed that the body attribute holds as much of the text as would fit in the space allocated for the attribute BODY. If the entire message text could not be contained in the alloted space then the mail message would be continued in another table. The deviation is being made in order to enhance access control over the objects of the text.

ID	TEXT

Figure 4.13. Body Table schema

The Body Table is envisioned to be a simple text table. The schema of the Body Table is illustrated in Figure 4.13 and the text attribute is described by the Schema Table tuple in Figure 4.14. TEXT follows a typical 80 column terminal screen format.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	TEXT	CHAR	80		BODY_TABLE

Figure 4.14. Schema Table rows for Body Table

G. DIRECTORY TABLE ACCESS TABLE

As noted earlier, the Directory Table Access Table is an access list which specifies those users authorized read access to the corresponding Mail Directory Table tuple and

access to the associated Mail Data Table. It should be noted that while the Directory Table Access Table filters access to the associated Mail Data Table, the user requesting access must still go through individual mail message checks involving the Mail Message Control Table, Access Control Table, and the Object Control Table before he/she is granted access to the individual tuples of the Mail Data Table.

ID	USER

Figure 4.15. Directory Table Access Table schema

In Figure 4.15 we see the basic structure of the Directory Table Access Table and in Figure 4.16 its Schema Table tuple is presented. Although the Directory Table Access Table is used as an access list, there are only implied pairs. The USER field is explicitly filled in with the userid of a user authorized access. The read capability for the corresponding Mail Directory Table tuple is implied. It is automatically assumed that the designated administrator of the associated Mail Data Table is allowed read access to the corresponding Mail Directory tuple and access to its Mail Data Table. His/her user-id need not appear in the Directory Table Access Table. The Directory Table Access Table may only be filled in by the designated administrator of the Mail Data Table or the SSO role.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	USER	CHAR	10	USERID	DIR_TBL_AC
				OWNER	ACCESS_TBL

Figure 4.16. Schema Table row for the Directory Table Access Table

H. MAIL MESSAGE CONTROL TABLE

The Mail Message Control Table is the first level of access control on an individual mail message basis. There is a row in the Mail Message Control table for each message in the associated Mail Data Table. Correspondence between the Mail Message Control Table and the Mail Data Table is maintained using the ID attribute. The ID of a tuple in the Mail Message Control Table is the same as its associated Mail Data Table tuple. The schema for the Mail Message Control Table is found in Figure 4.16 and the corresponding Schema Table tuples are presented in Figure 4.17.

ID	M_CLEAR	M_COMPART	M_CAVEAT	O_CLEAR

O_COMPART	O_CAVEAT	ACT	OCT

Figure 4.16. Mail Message Control Table schema

The M_CLEAR, M_COMPART, and M_CAVEAT fields correspond to the minimum clearance, compartments, and caveats that the user must be able to operate under in order to access the associated mail message. With regards to the MMS security

model, this is a modified form of the CONTAINER CLEARANCE attribute. The setting of these fields indicate a minimum clearance level that must be met in order to obtain access to the container. O_CLEAR, O_COMPART, and O_CAVEAT define the overall clearance, compartmentization, and caveat control of the corresponding mail message. ACT holds the unique name of a table holding the access control pairs for the mail message. OCT holds the unique name of the table which contains the classification of each object in the mail message.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	M_CLEAR	CHAR	1	CLEAR CLASS O_CLEAR	DIRECTORY MSG_CON_TBL
	M_COMPART	BOOLEAN	16	COMPART O_COMPART	DIRECTORY MSG_CON_TBL
	M_CAVEAT	BOOLEAN	16	CAVEAT O_CAVEAT	DIRECTORY MSG_CON_TBL
	O_CLEAR	CHAR	1	CLEAR M_CLEAR CLASS	MSG_CON_TBL
	O_COMPART	BOOLEAN	16	COMPART M_COMPART	MSG_CON_TBL
	O_CAVEAT	BOOLEAN	16	CAVEAT M_CAVEAT	MSG_CON_TBL
	ACT	CHAR	20		MSG_CON_TBL
	OCT	CHAR	20		MSG_CON_TBL

Figure 4.17. Schema Table rows for Mail Message Control Table

If the mail message was sent from another user, then the M_CLEAR, M_COMPART, and M_CAVEAT values are filled in by the system during the send operation from values indicated by the sender. Otherwise they are left blank until filled in by either the SSO role or the designated administrator. In the case of the administrator, he/she must meet any existing minimum requirements before he/she can change them. O_CLEAR, O_COMPART, O_CAVEAT, ACT, and OCT fields are filled in by the system during the send operation or a write operation.

I. ACCESS CONTROL TABLE

The Access Control Table holds the access pairs corresponding to the associated mail message. There is a tuple for each user or role authorized access to the mail message. Figure 4.19 gives the schema for the Access Control Table and Figure 4.20 shows the Schema Table tuples which describe it.

ID	USER	VIEWED_ACC	FROM_ACC	TO_ACC

COPY_TO_ACC	DTG_ACC	SUBJECT_ACC	BODY_ACC

Figure 4.19. Access Control Table schema

USER holds the designated role or the unique userid of the user granted access. Access is controlled on the basis

of NO ACCESS, READ ONLY, WRITE ONLY, or UPDATE (READ and WRITE). A two bit boolean is used to indicate the type of access allowed: 00 NO ACCESS, 01 READ ONLY, 10 WRITE ONLY, and 11 UPDATE. VIEWED_ACC, FROM_ACC, TO_ACC, COPY_TO_ACC, DTG_ACC, and SUBJECT_ACC indicate the respective fields of the Mail Data Table tuple to which they apply. With regard to Body_acc, this applies to the associated Body Table and not the BODY attribute. Only the designated administrator and the SSO role are allowed read and write access to the Access Control Table. Either may enter, delete, or modify access pairs in the Access Control Table. Each user indicated in the Access Control Table is allowed read only access to his/her respective tuple only in order to determine access rights.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	USER	CHAR	15	USERID OWNER	DIR_TBL_AC ACCESS_TBL
	VIEWED_ACC	BOOLEAN	2		ACCESS_TBL
	FROM_ACC	BOOLEAN	2		ACCESS_TBL
	TO_ACC	BOOLEAN	2		ACCESS_TBL
	COPY_TO_ACC	BOOLEAN	2		ACCESS_TBL
	DTG_ACC	BOOLEAN	2		ACCESS_TBL
	SUBJECT_ACC	BOOLEAN	2		ACCESS_TBL
	BODY_ACC	BOOLEAN	2		ACCESS_TBL

Figure 4.20. Schema Table rows for Access Control Table

J. OBJECT CONTROL TABLE

Since a mail message is a multiobject container, the Object Control Table is used to hold the classification, compartments, and caveats associated with each object in the corresponding mail message. Figure 4.21 gives the schema for the Object Control Table and the respective Schema Table tuples are presented in Figure 4.22. CLASS, COMPART, and CAVEAT represent the classification, compartments, and caveats assigned to the respective object. OFFSET is used to determine the delineation of objects in the body of the mail message as indicated below.

ID	CLASS	COMPART	CAVEAT	OFFSET

Figure 4.21. Object Control Table schema

The VIEWED, FROM, TO, COPY_TO, DTG, and SUBJECT attributes of a mail message tuple are each considered as single objects. The actual body of the mail message (found in the Body Table named in the BODY attribute) may or may not contain multiple objects. The VIEWED and DTG values will be looked upon as being unclassified and no tuples will be maintained in the Object Control Table for them. The first four rows of the Object Control Table are assigned the classification requirements of the TO, FROM, COPY_TO, and SUBJECT data respectively. The corresponding OFFSET values

for these objects indicate the length of the objects in characters.

Rows five onward indicate the respective classification requirements of the objects of the mail message body in the order that they appear in the body. The OFFSET value of a mail message body's object indicates the ID of the last row occupied by the object in the associated Body Table. The first object of the Body Table occupies rows one through its indicated OFFSET. Each subsequent object is delineated upon the previous object's OFFSET and its own OFFSET. Thus, each subsequent object occupies those rows from the next row past the previous object (ID = the previous object's OFFSET + 1) to its own OFFSET. This does require that the system provides one-up ID numbers at each send and write operations.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	CLASS	CHAR	1	CLEAR M_CLEAR O_CLEAR	OBJ_CONTRL
	COMPART	BOOLEAN	16	M_COMPART O_COMPART	USERS OBJ_CONTRL
	CAVEAT	BOOLEAN	16	M_CAVEAT O_CAVEAT	USERS OBJ_CONTRL
	OFFSET	INTEGER	6	ID	OBJ_CONTRL

Figure 4.22. Schema Table rows for Object Control Table

The Object Control Table is filled in by the system when the mail message is first created (written to a Mail Data Table by a send or write operation) and as part of each subsequent write operation. As one of the final functions of both the send and write operations, the O_CLEAR, O_COMPART, and O_CAVEAT fields of the Mail Message Control Table are filled in. This presumes that during these operations the system is able to determine the appropriate classification, compartments, and caveats associated with each object. There are a number of ways that this can be accomplished. For purpose of simplicity, it might be designated that the classification, compartments, and caveats for a given object must be in certain positions with respect to an object. In this case, the positional requirements might follow along with DoD requirements for classification markings of documents. For delineation of objects in the Body Table it might be required that all objects start with a TEXT tuple with the word "object" in its first six positions and the remaining positions blank. It is felt, however, that it would be placing undue implementation restrictions on the design if this thesis were to establish explicit means for conveying such information. Therefore, for purposes of this thesis, it will be assumed that a means has been established for conveying this information.

V. APPLICATION MODULARIZATION

A. INTRODUCTION

The aim of this chapter is to establish an appropriate modularization of information and functions which supports and enhances the proposed multilevel security aspects of the mail application. Section B gives a brief overview of the proposed modularization. Sections C, D, and E provide more in-depth information on the three principal modules. In order to more fully accommodate the MMS security model within the proposed modularization, some additional security features are necessary. These are discussed in section F. It should be noted that, as in the Military Message Systems security model, auditing is not addressed here.

B. OVERVIEW

Due to its general applicability, the modularization proposed in this chapter closely follows that proposed for MULTISAFE [Ref. 16]. The information and functions of the mail application are divided among three logically separated modules: Security Access Module (SAM), Storage and Retrieval Module (SRM), and the User Terminal Module (UTM). The SAM mediates all accesses to circumscribed information. Storage and retrieval of Mail Data Tables and Body Tables are performed by the SRM. The role of the UTM is to provide data manipulation functions and preprocess access queries. An

underlying assumption is made that each module has its own separate processing unit.

All communications between modules is by messages and only two logical paths of communications exist: UTM <--> SAM and SAM <--> SRM. The SAM, therefore, logically sits between the user and the mail messages. From this position it can control all accesses to circumscribed information.

As a quick reference for the reader, Figure 5.1 shows the distribution of the existing tables between the SAM and SRM modules. An additional table, the UTM Table, is described in section F. It is used for conveying the response to a user access request to the UTM.

SAM	SRM
SCHEMA	MAIL DATA
USERS	BODY
TRANS_ROLE	
TRANS_COMPART	
TRANS_CAVEAT	
MAIL DIRECTORY	
DIRECTORY TABLE ACCESS	
MAIL MESSAGE CONTROL	
ACCESS CONTROL	
OBJECT CONTROL	

Figure 5.1. Distribution of Tables

C. SECURITY ACCESS MODULE

As indicated above, the overall function of the SAM is to mediate all accesses to circumscribed information on the system. For the purposes of the application at hand, there are two types of circumscribed information. The first

includes the mail messages. The mail messages are the repository of the actual classified information which requires multilevel security. This information is found in the tuples of the Mail Data Tables and the associated Body Tables.

The second type of circumscribed information encompasses the access control information which the SAM needs to mediate access to the first type of information. A given mail message and its security control information define a relation which is normally considered integrally. Under this relation, a mail message and all of its associated security control information could theoretically be stored in the same tuple. In this case we would see the access control information described in Chapter 4 stored along with the associated mail message it protects. For example, such information as a given message's access pairs from the Access Control Table, the minimum clearance level required, and the overall classification level would be stored with the message. This could prove cumbersome and detract from the overall security of the system.

A special effort has been made, however, to maintain the separation of the message and its control information while sustaining the original relation. Given any control information as described in Chapter 4, it can be associated with the message it protects. Although this effort has resulted in the security control information of a given

message being distributed among a number of tables, it permits the system to take advantage of the fixed format of these tables in retrieving control information and enhances the security of the control information.

With the separation of the security control information from its associated classified information, a database of security information can be established which is strictly under the control of and accessible only by the SAM. This will encapsulate all of the access control related information within the SAM, enhancing the conditions for possible future verification by eliminating the dispersion of the access control information throughout the entire system.

At first it may appear that the Schema Table does not hold security information. In truth there is no direct security control relation between it and any given mail message. There is, however, an indirect relation since the Schema Table defines all other tables to the system. Because access control depends on accurately interpreting the information found in the other tables, the Schema Table plays a role in access control.

As was noted earlier, the SAM logically sits between the user and the stored mail messages. This is derived from its relative position with respect to the two logical communication paths. The existence of two logical paths is taken from the point of view of servicing a single user.

The user's request for access travels to the SAM via the UTM <--> SAM logical path. There the SAM determines if the access is allowed. If the access is allowed and requires retrieval by the SRM, then the SAM sends a request to the SRM via the SAM <--> SRM path for the required table or tables. When the SRM has performed the required retrieval, the table or tables are sent to the SAM via the SAM <--> SRM path. The SAM performs any filtering necessary and sends the response to the user via the UTM <--> SAM path.

The UTM <--> SAM and SAM <--> SRM paths are the only paths that a given user's request can travel. When viewed, however, with respect to the system as a whole, there is one SAM <--> SRM path, but there is a separate UTM <--> SAM path servicing each terminal accessing the system. To maintain the two logical paths concept for each given user, a unique user/terminal identifier must be appended in an unalterable manner to each request upon receipt by the SAM. The identifier remains with the request until it is answered by the SAM, thus assuring proper delivery back to the originating user. A similar concept is used by MULTISAFE [Ref. 16: pp. 390-394]. Such a method allows communication between users only via mail messages where one user's request sends a mail message to a second user's Mail Data Table and the second user's request reads the mail message. There is no direct communication between users and each user's request is handled as a separate message with no

inter-message communication. It is, therefore, reasonable to view the system as having only two logical communication paths.

As indicated above, the position of the SAM at the ends of the two logical communication paths allows it to mediate all accesses to circumscribed information. The first layer of the SAM's access mediation is its control over the flow of messages within the system. All message flows between the UTM and SAM and the SAM and SRM are controlled by SAM requests for message transmissions. The SAM polls the UTM's to determine if a user request exists. When a request is detected, the SAM acknowledges, allocates a buffer area for the request, requests the UTM to send the request, and assigns a user/terminal identifier to the request at time of transmission. Once the request is received in full, the SAM can begin processing it. The request must be received in full to insure that the user makes no changes to the request after access checking begins.

Processing starts with determining the type of request. If additional information is needed from the user, such as user authentication during the log on, the SAM issues a request to the UTM for the additional information. Once all necessary information is assembled, the SAM determines if the access is authorized according to its security control information. If the access is not authorized then the SAM sends a generic acknowledgement to indicate the access can

not be performed. A generic acknowledgement is sent in order to reduce possible covert channels of communication. Under such channels of communication, information can be conveyed by the type of denial acknowledgement made. If the access is authorized and access is to the security control information, the SAM retrieves the appropriate information from its local database, performs any required filtering of the information, attaches authenticator information to each tuple (explained in section E), and informs the UTM that a request response is available. When the UTM is ready, the SAM sends the response to the UTM.

If the user's request is authorized and requires service by the SRM, the SAM requests the SRM to retrieve the necessary tables. In much the same way it polls the UTM, the SAM polls the SRM to determine if it has any responses to requests ready for transmission. If a response is ready, the SAM acknowledges, allocates a buffer area for processing the response and requests the SRM to start transmission. Unlike processing the initial request, the entire response need not be received before the SAM begins processing it. During its processing of the response, the SAM filters the response based on its access control information, attaches authenticator information to each tuple (explained in section E), and informs the UTM that a request response is available. When the UTM is ready, the SAM sends the response to the UTM.

Within the description above for the SAM's control over the flow of messages, we have also seen its two other aspects of control associated with mediation of access. These are access authorization checking and filtering. Access authorization checking begins with the logon checking to guarantee the user is authorized access to the system. If access is authorized, then the SAM dynamically maintains user related information which will be necessary to mediate any accesses to circumscribed information. This would minimally include the user's id, user's current role, user's clearance level (to include clearance as well as compartment and caveat capabilities), and the maximum classification level of information that can be sent to the user.

The maximum classification level (to include classification as well as compartments and caveats) is determined by the maximum common clearance, compartment, and caveat values derived from the clearance levels (taken from the Users Table) of the user and the terminal he is operating from. For the maximum common classification, the minimum of the respective CLEAR values according to the DoD hierarchy is assumed. For the maximum common compartment and caveat values, the COMPART and CAVEAT values of the user clearance level are logically "AND"ed with the respective values of the terminal classification level. Taken together the maximum common classification, maximum common compartment, and the maximum common caveat values form the

maximum classification level of information that can be sent to the user.

For any access to circumscribed information, access authorization checking uses a layered approach where a user may be denied access at any layer in the checking. It should be noted that the layered approach presented here is based primarily on the underlying threading through of tables required by the distribution of control information and the basic serial nature of most processors. If parallel processing could be used in a verifiable manner, then the access checking could be done simultaneously.

Using access to information in a given tuple of a Mail Data Table or access to the associated Body Table as an example, access authorization checking begins with the Mail Directory Table. The SAM checks the minimum clearance, compartment, and caveat requirements associated with the corresponding Mail Data Table against the clearance level of the requesting user. If the user passes this layer, the SAM consults the Directory Table Access Table to determine if the user is on the access list for the associated Mail Data Table. Success here takes the SAM to the Mail Message Control Table. There the minimum clearance, compartment, and caveat requirements for the given mail message is checked against the clearance level of the requesting user. Upon success, the last layer of access authorization checking is reached. The Access Control Table is checked to determine

if the user or user's current role is on the access list along with permission for the requested access.

In the situation where the user is requesting access to access control information in the SAM's database, an abbreviated form of the above access authorization checking is used. The user's current role is checked. If the user is operating in the SSO role then the user will be granted access to the requested access control information which is not solely under the control of the system. If the requesting user is not operating under the SSO role, then the corresponding tuple of the Mail Directory Table is checked to determine if the user is the designated administrator (OWNER). Passing this layer, the requesting user must meet the minimum clearance, compartment, and caveat requirements stored in the Mail Directory Table tuple if accessing the Mail Directory Table or Directory Table Access Table control information. If the user is requesting access to tuples in the Mail Message Control Table or Access Control Table then he must meet the minimum clearance, compartment, and caveat requirements stored in the corresponding Mail Message Control Table. Provided the user meets the appropriate above requirements, then the user will be granted access to that control information which the administrator is authorized to as noted in Chapter 4.

A user who is not the designated administrator is allowed to read that information in the Access Control Table

which defines his/her access rights. In this case the user must meet the minimum clearance level indicated in the Mail Directory Table. If this is met then he/she must meet the minimum clearance level for the mail message that the Access Control Table is associated with. This minimum clearance level is indicated in the Mail Message Control Table. Finally, the user or user's current role must be included in the access list of the Access Control Table. Read only access is then granted to the access information associated with the requesting user.

Filtering is the final aspect of the SAM's access mediation discussed here. With filtering, the information to be included in a response to a given user's request is more exactly defined than is done in the more general access authorization checking. With regards to access of the access control information, filtering is incorporated partly in the access authorization checking. As described in Chapter 4, some of the attributes of those tables containing access control information are universally defined for system use only or for user access only under the SSO role. These attributes are automatically filtered out accordingly by the SAM.

With respect to the mail messages, filtering is based directly on the associated Access Control Table and Object Control Table. The Access Control Table is used to filter information based on discretionary access rights and the

Object Control Table is used to filter information based on mandatory clearance controls. Under mandatory clearance checking, the classification (taken from the Object Control Table) of each object not filtered out by the Access Control Table is checked to determine if it is less than or equal to the maximum classification level.

The object's classification is compared against the maximum common classification level. If it is less than or equal then the the maximum common compartment and caveat values are "XOR"ed (exclusive "OR"ed) with the object's respective values. This will eliminate all "1" bits that are common. The resultant compartment and caveat values are then "AND"ed with the object's respective values. If the object's classification level has any compartment or caveat values not included in the maximum classification level, then they will remain as "1" bits in the compartment and caveat values resulting from the "AND" operation. The information in the object is not forwarded to the user if its classification is greater than the maximum common classification or if either of the compartment or caveat values resulting from the "AND" operation are non-zero.

D. USER TERMINAL MODULE

As noted in the overview, the primary jobs of the UTM are to provide data manipulation functions and preprocess user access queries. The UTM derives its name from the

basic concept that all of the UTM software resides (minimally while the user is logged onto the system) in the sealed terminal, alterable only by authorized individuals. There are a number of ways that this might be done. One possible way is that the software resides in the ROM of the terminal. Another is that the software resides on nonvolatile internal secondary storage such as a Winchester disk or bubble memory. The software could also be downloaded to the terminal each time a user logs onto the system. Each has its associated costs.

In the case of the ROM, changes to the software requires new ROMs or reprogrammed PROMs. For the secondary memory, changes to software means going in and rewriting the storage. Downloading requires communication facilities which would allow rapid transfer of the software. All three methods would require sufficient RAM storage or secondary storage for application specific operations.

This application has been designed around the concept that there will be communication facilities for the rapid transfer of large amounts of data. All three basic access operations (read, write, and send) provide for the bulk transfer of data without manipulation. Accordingly, there is already a requirement for communication facilities which allow the rapid transfer of data. Thus, for the purposes of this thesis, it is assumed that the UTM software is

downloaded into RAM storage and that there is sufficient RAM storage to handle table storage and data manipulations.

Such a configuration would be more versatile than the other two cited, providing the user with the latest software at each logon. With this configuration, other applications of the original ISS system could be more easily incorporated by requesting the downloading of the appropriate software. It would even be possible to incorporate other applications not envisioned in ISS. Since security control is over objects, any application where the information could be divided into individual objects would be a candidate for implementation within this configuration. For example, the set of coordinates of a screen display or partial screen display could be considered as an object. Downloading graphics software to handle such sets of coordinates would allow the control and display of non-textual material, thus increasing the versatility of terminal usage and security software.

Unlike most modern multiuser systems where all the functions of a mail application are most likely found in a single module, the functions of the mail application of this thesis are distributed across the three modules in order to enhance security and minimize the impact on performance which often accompanies a high level of security. Security is enhanced because all of the access control information and security checking falls under one module. The impact on

performance is minimized because more simultaneity of operation is introduced. With separate processing units, each module performs independent of what the other modules are doing. In addition, the SAM is responsible only for security functions instead of security functions and user process execution like the CPUs of many multiuser computer systems.

Under this distribution of functions the actual physical storage and retrieval of mail messages is controlled by the SRM. The logical storage and retrieval, however, is controlled by the SAM. The SAM logically controls the storage and retrieval in the sense that it determines which Mail Data Table and Body Table the message is to be stored in or retrieved from and directs the SRM to perform the appropriate operation. The name of the Mail Data Table and Body Table are included as parameters in the SAM's request to the SRM for a storage or retrieval.

The logical storage and retrieval operations controlled by the SAM are read, send, and write. The handling of the read operation depends on whether it involves access control information or mail message information. If it involves access control information but the control information does not exist, the read is rejected. If the control message exists and the user is either operating in the SSO role or is the administrator of the associated Mail Data Table, then the read is authorized. If the user is requesting to read

his/her access rights for a given message and the user or user's role is in the Access Control Table for that message, then the read is authorized. All other read requests to control information are denied. For all authorized control information read requests, the control information is retrieved from the SAM's local database, filtered based on the restrictions noted in Chapter 4, and forwarded to the UTM.

In the case of mail message information, if the mail message does not exist, then the read is rejected. If the mail message exists, then the SAM checks the associated Access Control Table to determine whether the user or user's current role is listed with read privileges. If either is then the SAM requests the SRM to retrieve the mail message, filters the mail message based on control information from the Access Control Table and Object Control Table, and forwards the result to the UTM. Otherwise the read is denied.

For a send request the SAM determines if the target Mail Data Table exists. If it does not then the send is rejected. If it exists then the SAM checks the access list (Directory Table Access Table) for the destination Mail Data Table to determine whether the sender is in the access list. If the sender is on the list then the SAM appends the DTG, establishes the appropriate access control information,

and directs the SRM to store the mail message. Otherwise the send request is denied.

The write operation is handled is a manner similar to the read request. If the write involves access control information and the indicated control table does not exist, then the write is rejected. Otherwise the write access depends on whether the user is operating under the SSO role or is the administrator of the associated Mail Data Table. If either is the case then the write is authorized subject to the restrictions noted in Chapter 4. If not, the write is denied.

If the write involves mail message information and the indicated mail message does not exist, then the user is attempting to create a message through a write operation. Provided the user is requesting to have the information written to a Mail Data Table for which he/she is an administrator, the SAM fills in the DTG, establishes the appropriate access control information, and directs the SRM to store the mail message. Else the write is denied.

In the case where the mail message exists, the SAM determines which parts of the mail message (TO, FROM, COPY_TO, SUBJECT, text) that the user wishes to write. If the user or the user's current role is in the Access Control Table along with write permission for all of the requested parts or the user is the administrator of the associated Mail Data Table, then the SAM requests the SRM to retrieve

the associated mail message, overwrites the requested parts, updates the DTG field if blank, reestablishes the appropriate access control information, and requests the SRM to store the written mail message. Otherwise the write action is denied. In all cases of denied access operations, the user is notified appropriately.

With the logical read, send, and write operations (access requests) handled by the SAM and the actual physical storage of mail messages performed by the SRM, data manipulation and preprocessing of access requests are the primary jobs of the UTM. Data manipulation includes such actions as searching, displaying, and modifying retrieval information as well as the creation of new information. Much of the data manipulation operations, if not all, could be provided by the kernel of operators provided by ISS as discussed in chapter 3. The resident software of the UTM would provide these data manipulation capabilities.

As far as the preprocessing of the user access queries (read, send, and write), the resident software would assure that the requests are in the format required by the SAM. This could be done in a number of ways. One such way is that the resident software provides an interactive process which solicits the information to be included in the read, send, and write requests. Inherent in the data manipulation of retrieved data, creation of new data, and the preprocessing of access requests is the underlying memory

management and display of the highest classification level on the screen.

E. STORAGE AND RETRIEVAL MODULE

As indicated above, the primary functions of the SRM are the actual physical storage and retrieval of mail message information. These storage and retrieval operations occur in response to SAM requests. In order to provide for the integrity of the mail messages, the SRM also carries on backup operations on a periodic basis.

In order to provide for an undo capability, the SRM could incorporate an archival system which would maintain a set number of generations of a given mail message. Under such a system, each write operation causes a copy of the old version to be archived prior to the storing of the new version. If the user found that he wanted to go back to a previous version, he would request to read the desired archive version of the mail message and then write it as the current version.

F. ADDITIONAL SECURITY FEATURES

While the proposed modularization does effect certain benefits as noted above, it generates a gap in the security control of information which must be bridged. This gap is the lack of control over the information once it leaves the SAM either for the SRM or UTM. This lack of control occurs in two primary areas: the unauthorized manipulation of data

while in the UTM and SRM and the unauthorized reading of data while in the SRM module. The threat of unauthorized reading of data while in the UTM does not exist since the user only receives what he is authorized to see.

As noted in the preface to the security assumptions made in the MMS security model, "It will always be possible for a valid user to compromise information to which he has legitimate access [Ref. 15: p. 9]." Although compromise may always be a possibility, it is also a possibility to limit the range of means available to the user to effect the compromise. Without the access control information that the SAM has, it is impossible to control all unauthorized manipulation of data on the UTM's terminal screen. It is, however, possible to limit the screen oriented manipulations and to detect the changes if the data is written back to its respective tables or sent to an output device (assume output must go through SAM). Since no data manipulation is authorized while stored on the SRM, any change there would have to be detectable.

The procedure to accomplish this involves the use of additional attributes to accompany Mail Data Table and Body Table information. These attributes are authenticators and bits which indicate whether the associated information is read only. An authenticator is a means to provide an integrity check over a specified amount of data. It is a bit pattern which results from a calculation performed over

the data it is to check. The same bit pattern results each time the calculation is performed over the same data. If the data changes, however, a different bit pattern is produced by the calculation.

The number of authenticators which must be provided depends on the granularity of detection desired. In general, a separate authenticator would be appended to each tuple in order to detect any changes to the tuple as a whole. If, however, subsections of the tuple must be accounted for individually, then there would be an authenticator appended to the tuple for each subsection requiring an integrity check.

If we consider the integrity checking of a tuple from the Mail Data Table, eleven authenticators would be needed. An authenticator would be needed for each of the classifications of the classified objects (TO, FROM, COPY_TO, and SUBJECT), for each of the classified objects, and for the three unclassified data items (VIEWED, DTG, and BODY). Such an authenticator scheme would permit the detection of changes to the classification and to the values of the attributes.

While the configuration of the Mail Data Table tuple as described in Chapter 4 is convenient for the SAM's mediation of access and such an authenticator scheme would be convenient for data stored on the SRM, they are not necessarily convenient for the UTM. Neither do they provide

for easy detection of the case where a user inserts classified data into an object classified lower than the inserted data.

In order to overcome these weaknesses, a fixed table structure will be used in transferring data between the SAM and the UTM. Figure 5.2 shows the schema for the UTM Table and Figure 5.3 shows the respective Schema Table rows. R_ONLY indicates if the associated text is read only and can not be modified by the user. It is set by the SAM according to the associated value taken from the corresponding Access Control Table. C_AUTHEN and T_AUTHEN are authenticators for the classification and text of the tuple respectively. They are set by the SAM following separate authenticator calculations on the respective classification and text values. TYPE indicates the type of information in the text portion of the table. For example the following codes might be used to indicate the text is from the respective attributes of a Mail Data Table: T for TO, F for FROM, C for COPY_TO, V for VIEWED, D for DTG, and S for SUBJECT. CLASS, COMPART, and CAVEAT refer to the classification, compartments, and caveats associated with the text of the tuple. Their values are taken directly from the Object Control Table.

Finally, TEXT represents the information from the associated attributes. As described in Chapter 4, TO, FROM, COPY_TO, and SUBJECT are variable length fields. In order

to accommodate them in a fixed length TEXT column in the UTM Table, a decision must be made on how much to put in each TEXT column. If we assume that all data for these attributes originated from the UTM using lines from an 80 column screen to fill the TEXT portion of the UTM Table tuples, then the decision is made by the user. When the data is to be stored, the SAM need only use delimiters in the above variable length fields of the Mail Data Table to indicate where to split the variable length fields into fixed length TEXT fields format of the UTM Table when retrieved. This can also facilitate efficiency of physical storage on the SRM if all trailing blanks are suppressed by the SAM prior to storage.

ID	R_ONLY	C_AUTHEN	T_AUTHEN	CLASS	COMPART

CAVEAT	TYPE	TEXT

Figure 5.2. UTM Table

If we look at the other two types of information which are controlled by the SAM, the text of mail messages and access control information, we see that they can easily be accommodated by the UTM Table. Since the TEXT attribute of the Body Table is already stored in 80 character text format, the conversion to the UTM Table TEXT is direct. The

character B could be used to indicate the type and the remaining information could be set in the manner described above.

ID	NAME	TYPE	WIDTH	SYNONYM	TABLE
	R_ONLY	BOOLEAN	1		UTM_TABLE
	C_AUTHEN	BINARY	16		UTM_TABLE
	T_AUTHEN	BINARY	16		UTM_TABLE
	TYPE	CHAR	1		UTM_TABLE
	CLASS	CHAR	1		UTM_TABLE OBJ_CONTRL
	COMPART	BOOLEAN	16	M_COMPART O_COMPART	UTM_TABLE OBJ_CONTRL USERS
	CAVEAT	BOOLEAN	16	M_CAVEAT O_CAVEAT	UTM_TABLE OBJ_CONTRL
	TEXT	CHAR	80		UTM_TABLE BODY_TABLE

Figure 5.2. Schema Table rows for UTM Table

For control information, the values of control attributes could be held in the TEXT portion of the tuple, using delimiters to separate them. The TYPE field could indicate how to interpret the access control values in the TEXT field. For example, an M in the TYPE field might mean the values in the text field represent the minimum clearance, compartment, and caveat values from a tuple in the Mail Directory Table.

With the UTM Table and the underlying assumption that the user only has access to the TEXT information as presented by the UTM, the lack of control over unauthorized manipulation of information after it leaves the SAM for the UTM can be overcome. With the R_ONLY information, the UTM can prevent direct manipulation of read only material. The classification level values (CLASS, COMPART, and CAVEAT) along with the type can be used to prevent the insertion of higher classified information into objects of lower classification. Although the UTM performs these controls, they are only a form of security screening meant to limit the range of means available to the user to effect a compromise. They can limit to an extent what the user can display on a terminal to effect a compromise. The true security control still lies with the SAM.

With its access control information and the authenticators attached to the tuples, it can determine if any manipulations have taken place when the data is written back to a table or sent to an output device. To verify the data protected by an authenticator, the SAM needs only to recompute the authenticator based on the data returned. If the newly computed authenticator does not match the corresponding authenticator returned with the data, then a change has been made. Otherwise the data is assumed unmodified. Even if a change is made, it need not be unauthorized. The SAM would have to determine this from its

access control information and the operation being requested.

In the case of data stored on the SRM, no changes to the stored data are allowed. The major concern, therefore, is the detection of any changes. This would involve only the use of authenticators. Although the eleven authenticators per tuple scheme offered earlier would obtain fine granularity in detecting where changes have occurred in a Mail Data Table tuple, only one authenticator per tuple would be necessary to detect a change in the tuple. Although this only provides tuple level granularity in detecting where the change has occurred, it is probably justifiable and sufficient for most situations. Likewise, one authenticator per Body Table tuple would be sufficient. The checking for alterations would occur when the information is retrieved and follows the same type of authenticator check noted above.

The remaining control gap to be bridged is the unauthorized reading of data while stored on the SRM. The solution to this would be for the SAM to encrypt the mail message data before transmission to the SRM. Only those parameters required by the SRM for proper storage of the data need remain unencrypted.

Since the delineation of objects is maintained in the Object Control Tables, an interesting alternative solution to the "pass through" problem is possible. The "pass

through" problem is described as follows:

> The pass-through problem occurs when the database management system, in order to get to certain data, must access some other data which have different protection requirements. The situation is critical if these latter protection requirements are more stringent than the requirements for the requested data. An example of the problem is to search for confidential documents by passing through a pile of classified documents with information being designated as top secret, secret, and confidential. In this case, highly classified documents with top-secret and secret designations are being looked at for the purpose of finding the more lowly classified, confidential documents. A goal of every designer and implementor is to build secure database systems which will incur no pass-through problem. [Ref. 17: p. 233]

The "pass through" problem occurs in the SAM's filtering process. For example, if a user with a confidential clearance would be granted access to the text of a mail message with data classified up to top secret, the whole corresponding Body Table would be retrieved from the SRM in order to filter out all objects classified less than or equal to confidential.

Although the retrieval of the Body Table is unavoidable, it is still possible to minimize the "pass through" effect. If each object were encrypted with a key based in part on its classification, then from the Body Table example above, only the objects passing the filtering based on the Object Control Table need be unencrypted by the SAM. Thus, even though the other objects were effectively passed through by the SAM, their associated data would be unintelligible due to encryption.

VI. USER FUNCTIONS AND MODEL COMPLIANCE

A. INTRODUCTION

Two basic topics are covered in this chapter. Section B proposes the fundamental user functions which must be incorporated into the User Terminal Module (UTM) resident software so that the user can direct the Security Access Module (SAM) in its access mediation. Section C discusses the design's compliance with the security assertions of the Military Message System (MMS) security model.

B. USER FUNCTIONS

As indicated in Chapter 3, the aim of this thesis is to develop the conceptual design of those features which would make the electronic mail application of the Integrated Software System (ISS) multilevel secure. All efforts have been made to maintain this attitude of a high level design. To this extent, the interface between the user and the SAM's access mediation will be discussed in terms of interface functions provided the user through the UTM instead of implementation specific syntax and semantics. The definition of the latter should be accomplished during the design of the UTM resident software.

One of the goals of the ISS was the design of a set of primitive table operators and general system commands which would form the kernel of the ISS. This so called kernel of

commands as seen in Chapter 3 is common to all five applications of the ISS. As a kernel, the command set provides a degree of commonality which allows the user to move from processing in one application to another with a minimum of mental reorientation.

In keeping with the kernel concept, the user interface functions necessary to direct the SAM in its access mediation have been limited to seven: LOGON, ROLE SET, READ, WRITE, DOWNGRADE, SEND, AND ERASE. It is felt that these seven functions would provide the necessary user directed control while minimizing the number of functionally diverse operations that the SAM would have to recognize and accommodate. This minimization of interface functions should allow a degree of minimization in the amount of software necessary to implement the SAM. If other applications of the ISS were to be incorporated, the selected seven functions would form a kernel of security functions which would be common to all incorporated applications. Whereas the SEND function may appear to be mail application specific, it could be used in general to transfer any given application's tables from one user to another in a manner which preserves access mediation.

As the seven functions are discussed below, it should be remembered that all communications between the UTM and the SAM are under the direction of the SAM. The UTM preprocesses the user's access request (interface function) and at the

next polling by the SAM, informs the SAM that an access request exists. Once the SAM permits the transfer, the UTM transmits the request to the SAM.

As part of each interface function, the UTM gathers the information required by the access request. This may be done through user created tables designated in the request command, solicited from the user via an interactive process, or a combination of the two. Once the UTM has all of the information, it is placed in a UTM Table in a form recognizable by the SAM, to include setting the TYPE field for each tuple. Finally, the UTM Table is integrated into the message format used for all communication. These general operations must be performed for each of the functions presented.

The first function is the LOGON function. As with most systems, the user initiates the LOGON function in order to gain access to the system. During the logon, the UTM establishes connection with the SAM and requests the initiation of the logon sequence. During the logon sequence, the UTM solicits the authentication information from the user and transmits it to the SAM. Under the assumption that the UTM's software is downloaded after a successful logon, the UTM must receive the software and load it into the proper location in memory or secondary storage. If the SAM rejects the authentication, the UTM notifies the user and resolicits if directed by the SAM. In order to

perform the LOGON function, the UTM must have sufficient resident nonvolatile software (preferably ROM) to initiate and carry on these operations.

The ROLE SET function is provided to allow the user to change the role that he/she is currently operating under. When a user successfully logs onto the system, his/her initial role is established as a plain user (no established role). If the user wishes to perform an operation that requires a particular role (System Security Officer (SSO) or Downgrader for example), then the user must change his/her role to meet the requirements of the operation.

The READ function allows the user to read circumscribed information. Since the SAM provides no data manipulation (to include searching and conditional selection of tuples), read operations are performed on a table level basis. All tuples of a table which pass the filtering process are forwarded to the requesting user for any manipulation. In order to direct the SAM to the appropriate table, the UTM must solicit the table's name from the user. Since the Schema, Users, Trans_role, Trans_compart, Trans_caveat, and Mail Directory Tables are unique tables, they may be referenced directly by a form of their type such as Schema, Users, Trans_role, Trans_compart, Trans_caveat, and Directory.

Since the remaining tables are not unique (multiple occurrences of each type), an extended name must be provided

to direct the SAM. In order to reference a particular Mail Data Table, Mail Message Control Table, or Directory Table Access Table, the extended name must include a reference to the applicable Mail Data Table and an indication of the table type. Since addresses, as described in Chapter 4, are used to deliver mail to a given Mail Data Table, it seems logical to use the address to indicate the applicable Mail Data Table. Thus, a read request to the author's Directory Table Access Table might include RWYATT to indicate the applicable Mail Data Table and DIRECTORY ACCESS to indicate the table type. In a similar manner, to access a given Body Table, or Access Control Table associated with a given mail message, the user would have to supply the address of the applicable Mail Data Table, the DTG of the associated message, and the table type. The Object Control Table is system controlled and is, therefore, not readable by a user.

Unlike the read operation, which is performed on the table level, the write operation is performed on the tuple level (except in the case of the Body Table). Thus, the WRITE function must direct the SAM to the right table and tuple. The same table addressing scheme used by the READ function can be used for the WRITE function. For each tuple of information to be written in the indicated table, a unique identifier must accompany the information to direct the writing to the proper tuple. It should be noted that

while the object itself may be changed during a write operation, the classification of the object remains the same. Figure 6.1 indicates a possible identifier to use in directing the SAM to a particular tuple in the given table type.

TABLE TYPE	IDENTIFIER
1. Schema Table	- NAME
2. Users Table	- USERID
3. Trans_role	- SYS_FORM
4. Trans_compart	- SYS_FORM
5. Trans_caveat	- SYS_FORM
6. Mail Directory Table	- address (non-attribute)
7. Mail Data Table	- DTG
8. Mail Message Control Table	- DTG
9. Directory Table Access Table	- USER
10. Access Control Table	- USER

Figure 6.1. Possible identifiers for locating tuples

Again, the Object Control Table is not accessible by the user. In the case of the Body Table, a write operation is performed on the table level and only if the user has access to all objects of the given Body Table. The reasoning behind this is that the meaning of an object in the body table may be taken in part from its context in relation to the other objects of the Body Table. Without knowledge of the full context, the user would not necessarily know how his/her written object would be interpreted nor would he/she necessarily be able to assess the true classification of his/her object when written.

The downgrade operation is treated as if it were a write operation with the exception that the existing classifications of written objects may be downgraded. The DOWNGRADE function therefore solicits the same type of information as the WRITE function.

For the SEND function the UTM must also solicit direction to the correct Mail Data Table. This would consist of the address as noted in Chapter 4. The UTM must determine from the user the minimum clearance level, if any, to be forwarded with the mail message.

The ERASE function acts on the tuple level and deletes the given tuple. Access checking is performed as if it were a write operation. Therefore, the UTM must solicit tuple level direction information for the SAM in much the same manner as with the WRITE function. Although the ERASE function directs the deletion of a tuple in a given table, the deletion may have far reaching effects. A tuple deleted from the Mail Data Table causes the deletion of the entire mail message (to include the associated Body Table), the corresponding tuple in the Mail Message Control Table, the associated Access Control Table, and the associated Object Control Table. The deletion of a Mail Directory Table tuple causes the deletion of the corresponding Mail Data Table, associated Body Tables, and all control tables associated with the Mail Data Table and the individual mail messages. Such a deletion may only be made under the role of the SSO.

C. COMPLIANCE WITH SECURITY MODEL ASSERTIONS

Ten security assertions are made by the MMS security model. These are detailed in Appendix A. Until now no attempt has been made to tie design features to the security assertions or vice versa. In this section these connections will be presented. It should be noted that for this application, the concept of container has been addressed at a logical level as opposed to a physical one. To this end, the individual mail messages have been considered as the only containers with regard to measures taken to meet the security assertions. The discussion of compliance with the security assertions of the MMS security model are, therefore, prefaced upon this concept of containers.

One may argue that Mail Data Tables should be considered as containers also. Indeed the Mail Data Tables do contain the classified messages in the physical sense, however, with two notable exceptions, all access to classified information is mediated based upon applying the MMS security assertions on the level of individual messages. The first exception is that the designated administrator may establish a minimum clearance level requirement to be met by a user before obtaining general access to the associated Mail Data Table. This allows the administrator to accommodate the situation where the relation formed when the given mail messages are gathered together requires a minimum level of classification. Although success at this layer of access

mediation allows general access to the Mail Data Table, it does not generate an access to any classified information. The user must still pass the access mediation required at the individual mail message level to receive any classified information.

The second exception is that the SSO role may delete a Mail Data Table, and therefore its associated classified data, by deleting the corresponding Mail Directory Table tuple. While this is not a direct access, it does effect an access to classified information.

1. Authorization

All accesses to individual mail messages are filtered based on the Access Control Table associated with the mail message. It lists each user or role authorized access to the given mail message and the respective authorized accesses. A user's access request will be performed only if the user's userid or current role is in the Access Control Table and he/she is authorized to perform the requested access.

2. Classification hierarchy

The overall classification level of each message is maintained in the corresponding Mail Message Control Table. This value is established at the creation of the message and reestablished after each subsequent write operation to the mail message. Although this establishes the actual overall classification of the mail message, the clearance level

requirement for the mail message may be established even higher through the associated minimum clearance level requirement also stored in the Mail Message Control Table.

3. <u>Changes to objects</u>

As indicated in Chapter 5, the classification associated with each object is attached to each tuple in the UTM Table conveying that object. Authenticators are used to guarantee the integrity of these classification markings. If the user attempts to insert other previously classified tuples of higher classification into an object of a lower classification, then the UTM should be able to prevent this. The final authority for detecting this, however, is the SAM.

4. <u>Viewing</u>

During the SAM's filtering process, the classification of each object not rejected by the access authorization checking is checked against the maximum classification level of data that can be sent to the user. This maximum classification level is determined from the maximum common values between the user's clearance level and the clearance level of the terminal that he is operating from. If the classification of the object exceeds the maximum classification level, then the object is not forwarded to the requesting user.

5. Viewing CCR entities

A slight deviation has been made from the "Container Clearance Required" concept presented in the MMS security model. Instead of limiting the capability of specifying the minimum level of clearance required for accessing a container to the clearance level of the container, the SSO or Owner may indicate the specific level of clearance which the user must meet for access. This includes levels of clearance which may be greater than or less than the actual clearance level of the container. These minimum requirements must be met regardless of whether the reference is made directly or indirectly.

6. Translating indirect references

In all cases, the requesting user must meet the Viewing and Viewing CCR Entities requirements as stated above in order to see the ID of a container.

7. Labeling requirement

This is a UTM implementation oriented requirement that has not been covered in the design presented here. The classification associated with each object sent from the SAM to the UTM is recorded with each tuple of that object. It is expected that the classification of each object is displayed with that object and that the overall classification of all objects simultaneously displayed on the terminal's screen is itself displayed appropriately at

the top and bottom of the screen, but no explicit provisions have been made for this.

8. Clearance setting

The clearance of each authorized user and terminal is stored in the Users Table. Only the SSO role may access this table.

9. Downgrading

The DOWNGRADING function has been provided as one of the kernel interface functions for the system. Only through this operation will the user be allowed to downgrade the existing classification of an object.

10. Releasing

Since the application presented here is electronic mail as opposed to a true message system, releasing has not been incorporated. If releasing were to be incorporated, it could be made into an application specific function and handled in a manner similar to the SEND function.

VII. CONCLUSIONS AND RECOMMENDATIONS

A. THESIS DEVELOPMENT

This thesis supports the conceptual design of a multilevel secure electronic mail application. Instead of developing the entire conceptual design, to include the design of application specific features as well as security features, an approach was taken which called for designing security features that would be integrated into an existing conceptual design for an electronic mail application. The existing conceptual design chosen was for the electronic mail application of the Integrated Software System (ISS). Thus, the central theme of this thesis has been the conceptual design of those security features which would permit a stand alone version of the ISS electronic mail application to run in a multilevel security mode.

The thesis can basically be broken down into two parts, each of which has its own subparts. In the first, a firm framework of terminology and ideas was developed through a systematic examination of the multilevel security issue and related work. During the examination of formal models and SIGMA, it became clear that the Bell-LaPadula model would not be appropriate for an electronic mail application. As a consequence, the Military Message System security model was chosen to guide the development of the necessary security

features. In a similar manner, the review of the MULTISAFE system pointed out the applicability of the general modularization principles fostered in the development of MULTISAFE. These principles became the foundation of the modularization used in this thesis.

In the second part, the conceptual design of the security features was developed. Its development followed along the lines of the three objectives laid out in the introduction. First, the attributes necessary to support the required access mediation were defined. The defining of the required attributes led to the identification of twelve table types to be used in supporting the access mediation and electronic mail application. With the twelve table types, it was possible to separate the access control information from the data it protected while maintaining all relations which existed between the two.

Under the second objective, a modularization of functions and information was developed. Following closely the modularization scheme proposed for MULTISAFE, the functions of the proposed multilevel secure electronic mail application were divided among three modules: Security Access Module (SAM), User Terminal Module (UTM), and Storage and Retrieval Module (SRM). Since the SAM acts as the mediator of all access requests, all of the access mediation functions were concentrated in it. The separation of the access control information from the data it protected, as

noted above, allowed all access control information to be resident in a database controlled solely by the SAM, thus enhancing the overall security of the system. The UTM was assigned the functions of preprocessing user generated access requests and data manipulation. The actual physical storage of the mail messages was bestowed upon the SRM.

It was discovered that the modularization did fall short in some areas of control once the data left the SAM for the UTM or SRM. Primarily the areas were the unauthorized manipulation of data while under the control of the UTM or SRM and the unauthorized reading of data while under the control of the SRM. Through the proposed use of authenticators, the unauthorized manipulation of data can be controlled to a great extent in both the UTM and the SRM. In the case of the UTM, an additional table, the UTM Table, was created to facilitate the transmission of data with authenticators to the UTM. Encryption before storage offers adequate protection against unauthorized reading of data while under the control of the SRM.

In the last objective, the user interface to the access mediator was defined. Maintaining the idea that the thesis is a high level design, the interface was described in terms of functions performed rather than explicit syntax and semantics. In keeping with the general philosophy of the ISS, the number of functions required was kept to a minimum. Seven functions were identified. These seven

functions could form a kernel of functions that would also serve any other ISS application if integrated with the proposed electronic mail application.

B. CONCLUSIONS

It is somewhat difficult to draw conclusions about the conceptual design presented here. At best, it presents a somewhat formalized train of thought. There are no scales of measurement against which it can be judged for goodness, completeness, or worth. At best one can say, "Well, it looks good, seems complete, and may have some value." Any actual measurements would have to wait until future stages of development. It is left to the reader to pronounce the judgement of whether the design presents sufficient merit to continue its development. What will be presented here are those merits which the author considers important and some thoughts on the method of development.

As mentioned above, there are no true scales of measurement for this conceptual design. One can, however, establish those parts of the design which are felt to comply with the security assumptions of the MMS security model. This has been done in Chapter 6. In review, it was shown that compliance can be established for seven of the ten assumptions. As far as the remaining three, each should be looked at separately. For "Viewing CCR entities", it is not a case of non-compliance, but one of a change of approach

which is felt to increase the flexibility of the intent behind the "Viewing CCR entities" assertion. If it is felt that the original intent should be implemented, the necessary attributes are present which would allow a rapid change. With regard to the "Labeling requirement", there is no compliance. This is not because noncompliance is intended, but rather due to the idea that compliance would be established at a future stage when it is decided exactly how to handle obtaining the classification of an object from the user. It is felt that that decision is too close to the implementation stage to be presented in this design. Finally, it is felt that the "Releasing" assertion does not properly apply to the electronic mail application but could be implemented if necessary. Thus, all of the assertions are accounted for, could be accounted for, or will be accounted for in a final implementation.

The modularization does pose the possibility of the absence or at least the minimization of certain possible problems. Although this is a multiuser system, there is virtually no way that one user can affect another except through authorized means of communication. For example, the transactional nature of the design would preclude the possibility of one user's process affecting another user's as is the case in many other multiuser systems. The likelihood of the existence and the range of effect of subversive user action, such as a Trojan Horse, would be

minimized. Since the underlying concept is that all UTM software used by the user is downloaded and not copied back, a Trojan Horse would have to use the same lines of communication as the user and go through the same access checking in order to get information back to another user. This should be easily detectable. In the case of Trojan Horses in the SRM software, all the classified data is encrypted making it useless to another user without the decryption key. The only area where a Trojan Horse might have a valuable effect is in the SAM. If the SAM is verified and adequately protected, the existence of a Trojan Horse there would be impossible.

As a final note on the proposed modularization of the electronic mail application, it should be noted that it is not required that each module be physically separated from the other modules. This is one possible implementation method but not the only one. Such a configuration might be useful in the situation where a central unit which services a number of simultaneous user workstations by mediating access to a common bank of secondary storage. Another possible situation is a stand alone single user workstation which services one user at a time but may service a number of users over a period of time. An example would be a word processor with a Winchester disk for secondary storage. In such a situation, the modules could be implemented as separate processor boards.

As a comment on using an existing conceptual design of an electronic mail application, it should be noted that the conceptual design of features to make it multilevel secure was made easier, but one must be aware of possible pitfalls. Like an actual implementation, an existing conceptual design can also introduce restrictions if one allows it. To an extent this is the case here, but probably on a much smaller scale. Initially too much emphasis was placed on maintaining some of existing design concepts. Eventually it was determined that the original design would have to be modified to accommodate the proposed security features. This led to some delay in developing the design and some probable inefficiency which may still exist. If it were to be done again, some changes would be made. As an example, it might be better to store mail messages entirely in fixed format Mail Data Tables. An appropriate format might be akin to that of the UTM Table.

C. RECOMMENDATIONS

As a general recommendation, it is felt that the development of the multilevel secure electronic mail application should be continued. In terms of specific recommendations, there are three. The first stems from experience developed during the conceptual design presented here. The development of multilevel security features is extremely complex. Many blind alleys were searched before

the features presented here were reached. It is strongly recommended that any future work on the design be done as a group effort instead of individual effort. This would probably eliminate many blind alleys or at least shorten them. With an existing base design, the work could be divided up, allowing more individual attention to details.

A heavy emphasis should be put on efficiency and parallelism of operation. At the conceptual level it is difficult to do this because there are no means of measurement. As the development continues, though, measurement should become possible. Since security does increase the overhead of operations, all attempts should be made to minimize its effect. Minimization, however, will have to be tempered by some type of analysis which will identify the point at which further efforts at minimization would no longer cost beneficial.

Finally, a serious examination should be made into storing all the mail messages of a given user in a single fixed format Mail Data Table as opposed to the presently proposed situation where mail messages are split between the Mail Data Table and the respective Body Tables. This would present more uniformity and allow easier adaptation of the proposed conceptual design to the incorporation of any other ISS application.

APPENDIX A

This Appendix depicts the security model of the Military Message System (MMS) through an excerpt from "Military Message Systems: Requirements and Security Model". [Ref. 15: pp. 6-10]

IV. SECURITY MODEL

The security model for the MMS family is intended to provide a framework for users to understand system security, to guide the design of each family member, and to provide a basis for certifiers to review the system. Although we intend to have a single security model for the entire MMS family, each member will require a separate security analysis. The model presented here is informal; we expect it to provide a basis for a more formal version that may be used as a basis for program verification efforts.

In this section we define some terms, use them to specify a model of how a user views the system's operation, and state assumptions and assertions, based on the terms and the model of operation, that are intended to be sufficient to assure the security of the system. The security model includes the definitions, user's view of operation, the assumptions, and the assertions. It is a revision of earlier work.

This model does not address auditing, although message systems clearly require auditing mechanisms. The existence of an audit trail may deter potential penetrators, but auditing is primarily a technique for detecting security violations after the fact. The security model focuses on assertions that, if correctly enforced, will prevent security violations. Consequently, assertions and assumptions about auditing do not appear; in a more detailed system specification, auditing requirements would be explicit.

Definitions

The definitions below correspond in most cases to those in general use and are given here simply to establish an

explicit basis for the model. We distinguish between "objects", which are single-level, and "containers", which are multilevel. We also introduce the concept of "user roles", which correspond to particular job-related sets of privileges.

Classification: a designation attached to information that reflects the damage that could be caused by unauthorized disclosure of that information. A classification includes a sensitivity level (UNCLASSIFIED, CONFIDENTIAL, SECRET, or TOP SECRET) and a set of zero or more compartments (NATO, NUCLEAR, etc.). The set of classifications, together with the relation defining the allowed information flows between levels, form a lattice. Most dissemination controls, such as NATO only, NOFORN, and NOCONTRACTOR, can be handled as additional compartment names.

Clearance: the degree of trust associated with a person. This is established on the basis of background investigations and the functions required of the individual. It is expressed in the same way as classifications are, as a sensitivity level and a (possibly null) compartment set. In a secure MMS, each user will have a clearance, and functions performed by the MMS for that user may check the user's clearance and the classifications of objects to be operated on. Some other characteristics of a user, such as his nationality and employer, may also be treated as part of his clearance so that dissemination controls are handled properly within this framework.

UserID: a character string used to denote a user of the system. To use the MMS, a person must present a userID to the system, and the system must authenticate that the user is the person corresponding to that userID. This procedure is called logging in. Since clearances are recorded on the basis of one per userID, each user should have a unique userID.

User: A person who is authorized to use the MMS.

Role: The job the user is performing, such as downgrader, releaser, distributor, etc. A user is always associated with at least one role at any instant, and the user can change roles during a session. To act in a given role, the user must be authorized for it. Some roles may be assumed by only one user at a time (e.g., distributor). With each role comes the ability to perform certain functions.

Object: an abstraction implemented by an MMS. An object is the smallest unit of information in the system to which a classification is explicitly attached. An object thus contains no other objects -- it is not multilevel. There are many kinds of objects: an example is the data-time-group of a message.

Container: an abstraction implemented by an MMS. A container has a classification and may contain objects (each with its own classification) and/or other containers. In most MMS family members, message files and messages are containers. Some fields of a message (such as the Text field) may be containers as well. The distinction between an object and a container is based on type, not current contents: within a family member, if an entity of type message file is a container, then all message files in that family member are containers, even if some of them are empty or contain only objects and/or containers classified at the same level as the message file itself. Devices such as disks, printers, tape drives, and users' terminals will be containers, rather than objects, in most MMS family members.

Entity: either a container or an object.

Container Clearance Required (CCR): an attribute of some containers. For some containers, it is important to require a minimum clearance, so that if a user does not have at least this clearance, he cannot view any of the entities within the container. Such containers are marked with the attribute "Container Clearance Required" (CCR). For example, a user with only a CONFIDENTIAL clearance could be prohibited from viewing just the CONFIDENTIAL paragraphs of a message classified TOP SECRET. On the other hand, given a message file containing both TOP SECRET and CONFIDENTIAL messages, it may be acceptable to allow the user in question to view the CONFIDENTIAL ones, even though the container (message file) as a while is classified TOP SECRET.

ID: identifier. An ID names an entity without referring to other entities. For example, the originator and date-time-group of a message constitute an ID for that message. Some, but not necessarily all, entities are named by identifiers. Entities may also be named in other ways, e.g., "the third paragraph in the text of the second message in the container INBOX."

Direct reference: a reference to an entity is direct if the entity's ID is used to name it.

Indirect reference: a reference to an entity is indirect if a sequence of two or more entity names (of which only the first may be an ID) is used to name it.

Operation: a function that can be applied to an entity. It may simply allow that entity to be viewed (e.g., display a message), or it may modify the entity (update a message), or both(create a message). Some functions may involve more that one entity (copy a message from one message file to another).

Access Set: a set of pairs (userID or role, operation) that is associated with an entity. The operations that may be specified for a particular entity depend on the type of that entity. For messages, operations include DISPLAY, UPDATE, DELETE, etc. The existence of a particular pair in the access set implies that the user corresponding to the specified userID or role is authorized to invoke the specified operation on the entity with which the set is associated.

Message: a particular type implemented by an MMS. In more MMS family members, a message will be a container, though messages may be objects in some receive-only systems. A message will include To, From, Date-Time-Group, Subject, and Text fields, and additional fields as well. A draft message also includes Drafter and Releaser fields.

User's View of MMS Operation

We present the following as a model of the use of a secure MMS. Terms defined above are printed in upper case.

People initiate use of the system by logging in. To log in, a person presents USERID and the system performs authentication, using passwords, fingerprint recognition, or any appropriate technique. Following a successful authentication, the USER invokes OPERATIONS to perform the functions of the message system. the OPERATIONS a USER may invoke depend on his USERID and his current ROLE; by applying OPERATIONS, the USER may view or modify OBJECTS or CONTAINERS. The system enforces the security assertions listed below (that is, it prevents the user from performing OPERATIONS that would contradict these assertions).

Security Assumptions

It will always be possible for a valid user to compromise information to which he has legitimate access. To make the dependence of system security on user behavior explicit, we list the following assumptions. These assumptions are really security assertions that can only be enforced by the users of the system.

A1. The System Security Officer (SSO) is assumed to assign clearances, device classifications, and roles properly.

A2. The user is assumed to enter the correct classification when composing, editing, or reclassifying information.

A3. Within a classification, the user is assumed to address messages and to define access sets for entities he creates so that only users with a valid need-to-know can view the information.

A4. The user is assumed to control properly information extracted from containers marked CCR (i.e., to exercise discretion in moving that information to entities that may not be marked CCR).

The basis for these assumptions is that when there is no other source of information about the classification of an entity or the clearance of a person, the user is assumed to provide information that is correct.

Security Assertions

The following statements are to be demonstrated to hold for a multilevel secure MMS:

Authorization	1.	A user can only invoke an operation on an entity if the user's userID or current role appears in the entity's access set along with that operation.
Classification hierarchy	2.	The classification of any container is always at least as high as the maximum of the classifications of the entities it contains.
Changes to objects	3.	Information removed from an object inherits the classification of that object. Information inserted into an

object must not be classified at a level above the classification of that object.

Viewing	4.	A user can only view (on some output medium) an entity with a classification less than or equal to the user's clearance and the classification of the output medium. (This assertion applies to entities referred to either directly or indirectly.)
Viewing CCR entities	5.	A user can view an indirectly referenced entity within a container marked "Container Clearance Required" only if the user's clearance is greater than or equal to the classification of that container.
Translating indirect references	6.	A user can obtain the ID for an entity that he has referred to indirectly only if he is authorized to view that entity via that reference.
Labeling requirement	7.	Any entity viewed by a user must be labelled with its classification.
Clearance-setting	8.	Only a user with the role of System Security Officer can set the clearance recorded for a userID.
Downgrading	9.	No classification marking can be downgraded except by a user with the role of downgrader who has invoked a downgrade operation.
Releasing	10.	No draft message can be released except by a user with the role of releaser. The userID of the releaser must be recorded in the "releaser" field of the draft message.

LIST OF REFERENCES

1. Harrison, P. J. and Thompson, G. L., Design of an Integrated Software System Based on the Relational Data Base Model, Master's Thesis, Naval Postgraduate School, December 1983.

2. Schell, R. R., "Computer Security the Achilles' Heel of the Electronic Air Force", Air University Review, January-February 1979.

3. Ralston, A. (Edited by), Encyclopedia of Computer Science, Van Nostrand Reinhold Company, 1976.

4. Stryker, D., Subversion of a Secure Operating System, Navy Research Laboratory, NRL Memorandum Report 2821, June 1974.

5. Tagney, J. D., History of Protection in Computer Systems, The MITRE Corporation, MTR-3999, 15 July 1980.

6. Ware, W. H. (Edited by), Security Controls for Computer Systems, RAND, Report of Defense Science Board Task Force on Computer Security, R-609, February 1970.

7. Chairman Computer Security Technical Consortium (Prepared by), "Department of Defense Computer Security Initiative: A Status Report and R & D Plan", Information Systems Directorate Assistant Secretary of Defense Communications, Command, Control, and Intelligence, March 1981.

8. "Trusted Computer System Evaluation Criteria" Final Draft, Department of Defense Computer Security Center, 27 January 1983.

9. "ADP Security Manual", Department of Defense, DoD 5200.28-M, January 1973.

10. Landwehr, C. E., "Formal Models for Computer Security", ACM Computing Surveys, Vol. 13, Number 3, September 1981.

11. Berg, H. K., Boebert, W. E., Franta, W. R., and Moher, T. G., Formal Methods of Program Verification and Specification, Prentice-Hall, Inc., 1982.

12. Committee on Multilevel Data Management Security, Air Force Studies Board, Commission on Engineering and Technical Systems, National Research Council,

"Multilevel Data Management Security", National Academy Press, 1983.

13. Ullman, J. D., Principles of Database Systems, Computer Science Press, 1982.

14. Heitmeyer, C. L. and Wilson, S. H., "Military Message Systems: Current Status and Future Directions", IEEE Transactions on Communications, Vol. Com-28, No. 9, September 1980.

15. Landwehr, C. E. and Heitmeyer, C. L., "Military Message Systems: Requirements and Security Model", Computer Science and Systems Branch, Information Technology Division, Naval Research Laboratory, NRL Memorandum Report 4925, 30 September 1982.

16. Trueblood, R. P., Hartson, H. R., and Martin, J. J., "MULTISAFE - A Modular Multiprocessing Approach to Secure Database Management", ACM Transactions on Database Systems, Vol. 8, No. 3, September 1983.

17. Hsiao, D. K, Kerr, D. S., and Madnick, S. E., Computer Security, Academic Press, 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93943	2
4.	Professor Dushan Z. Badal, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93943	2
5.	LCDR Alan K. Johnson, Code 52Jn Department of Computer Science Naval Postgraduate School Monterey, California 93943	2
6.	Curricular Office, Code 37 Computer Technology Naval Postgraduate School Monterey, California 93943	1
7.	DoD Computer Security Center Deputy Director Roger R. Schell, COL USAF 9800 Savage Road Fort George G. Meade, Maryland 20755	2
8.	DoD Computer Security Center Office of Research and Development George F. Jelen 9800 Savage Road Fort George G. Meade, Maryland 20755	1

9. DoD Computer Security Center 4
Office of Systems Applications
Robert W. Wyatt
9800 Savage Road
Fort George G. Meade, Maryland 20755